PE Practical Electronics

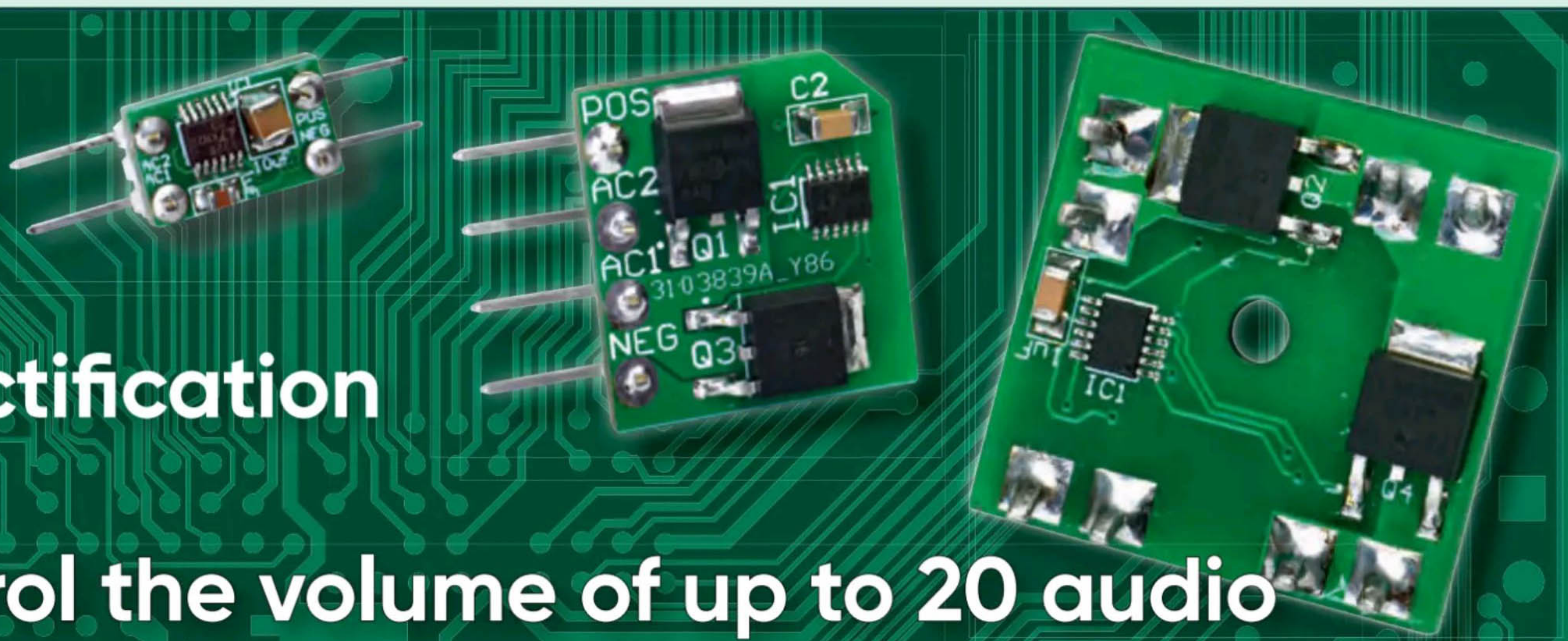
The UK's premier electronics, computing and maker magazine

Circuit Surgery
DAC frequency response correction

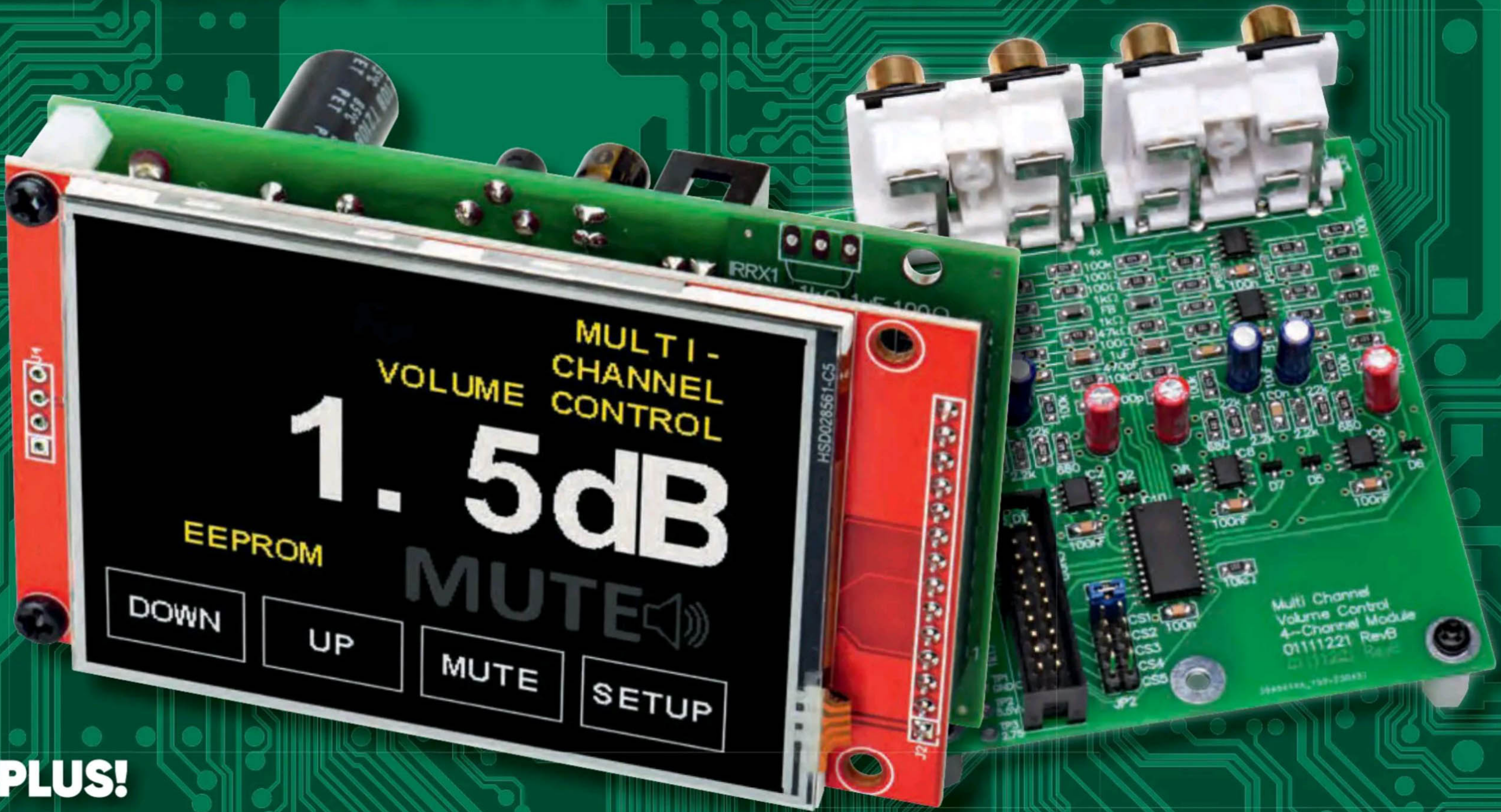
Teach-In 2024
Learn electronics with the ESP32, Part 9 – Putting it all together

Practically Speaking
Avoiding electric shocks during repairs

Six PCBs for highly efficient active rectification


POS
C2
AC2
AC1
Q1
IC1
3103839A_Y86
NEG
Q3
Q2
IC1
Q4

Control the volume of up to 20 audio channels with a touchscreen or remote!


MULTI-CHANNEL
VOLUME CONTROL
1.5dB
EEPROM
MUTE
DOWN
UP
MUTE
SETUP
IRRX1
Multi Channel Volume Control 4-Channel Module 01111221 RevB

PLUS!

MG4 XPower EV review by Julian Edgar

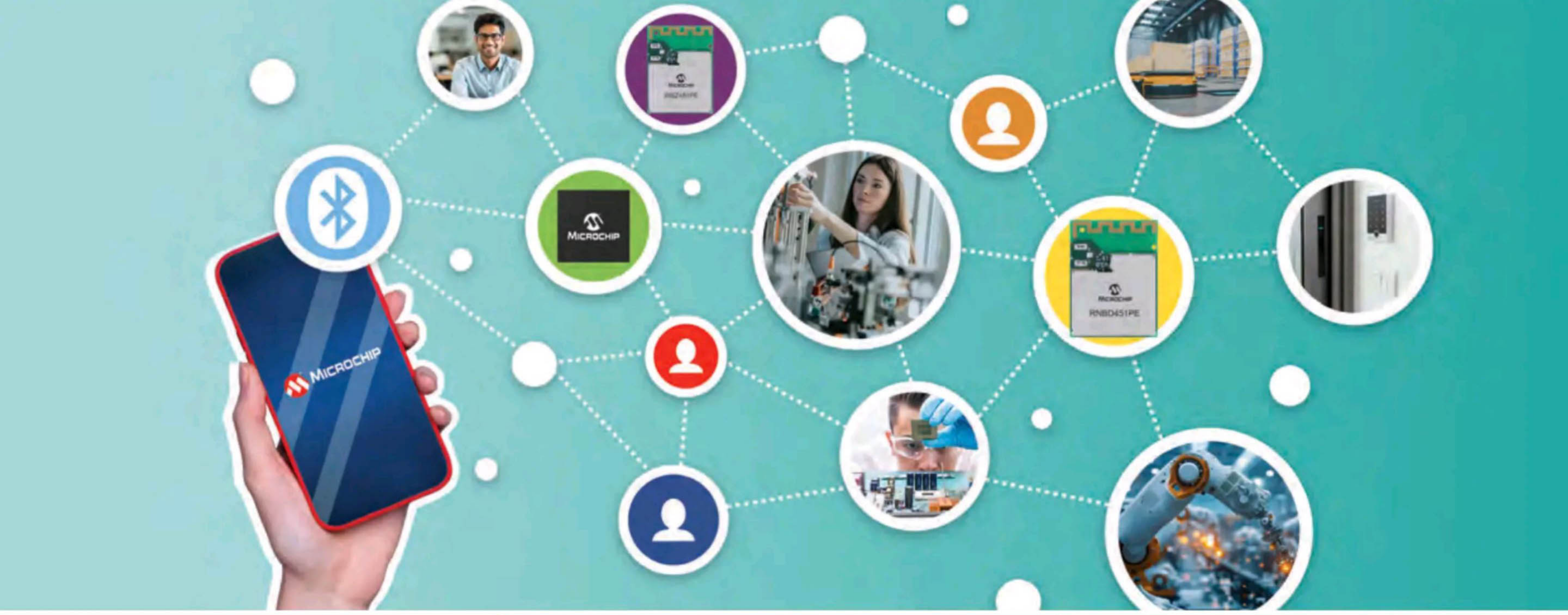
Techno Talk – On robots and ukuleles

Cool Beans – There's more than one way to blank zeros

Net Work – *PE*'s 60th anniversary and its history

Nov 2024 £5.99

www.electronpublishing.com | @practicalelec | practicalelectronics



The Power of Choice

Beginner or Expert, Bluetooth® Solutions for All

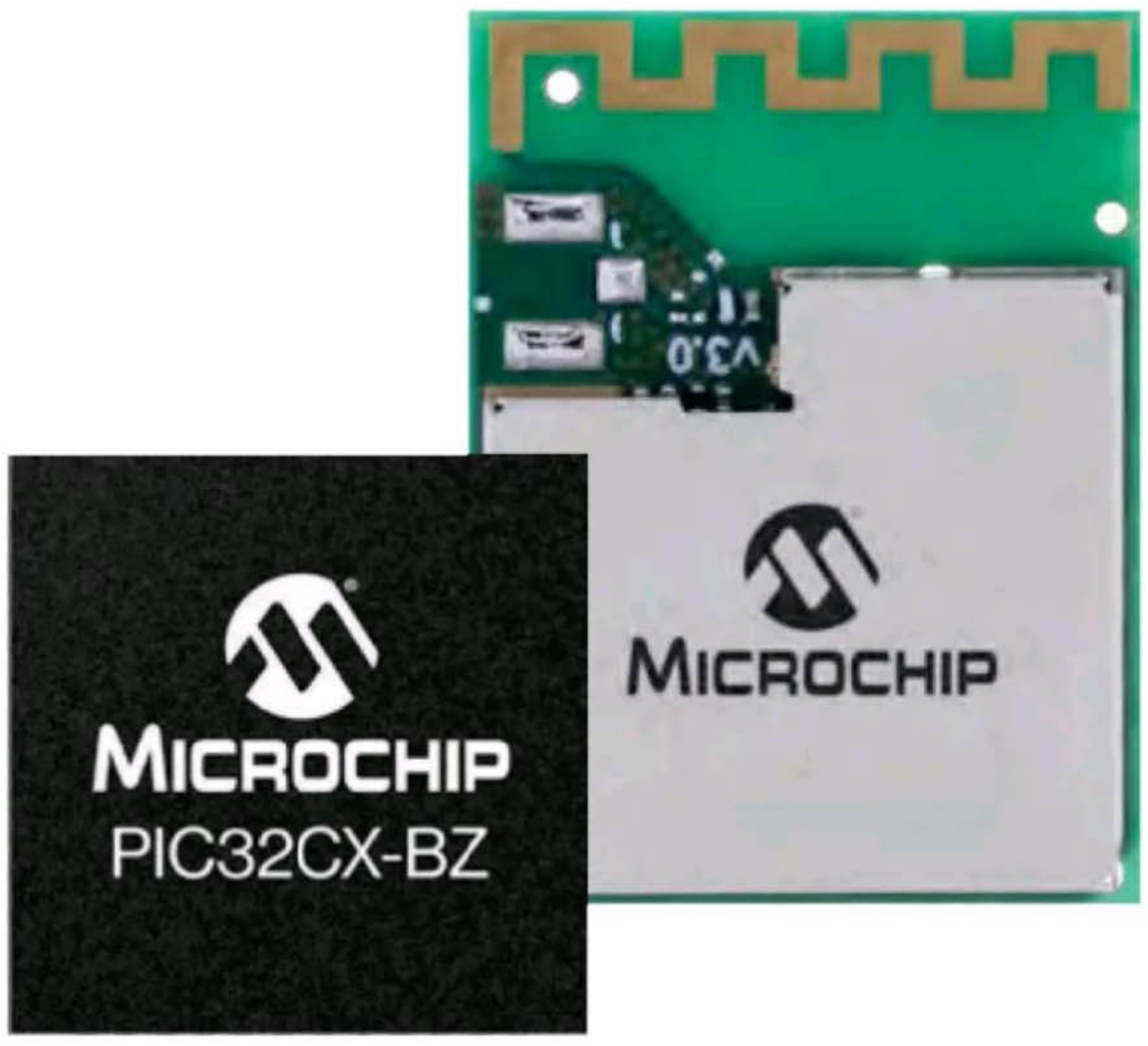
Adding Bluetooth to your product adds functionality while improving the user experience. However, finding the right provider can be difficult.

Most vendors offer an overwhelming number of products with little guidance to help you choose between them. There are so many choices that it is hard to feel confident in your selection.

Whether you are an experienced professional or new to Bluetooth development, we have a solution tailored to your needs. Our expanded Bluetooth product portfolio provides a range of options that are highly functional and user-friendly.

- **Plug-and-play modules** that require no programming or RF expertise
- **Programmable modules** that are regulatory certified multi-protocol MCUs with Bluetooth and 802.15.4
- **Robust SoCs** that are 125 °C qualified, peripheral rich and well suited for tough applications
- **Tools and support** to help you find the right part and quickly move from concept to prototype

Browse our portfolio to find the right certified Bluetooth device for you.



MICROCHIP

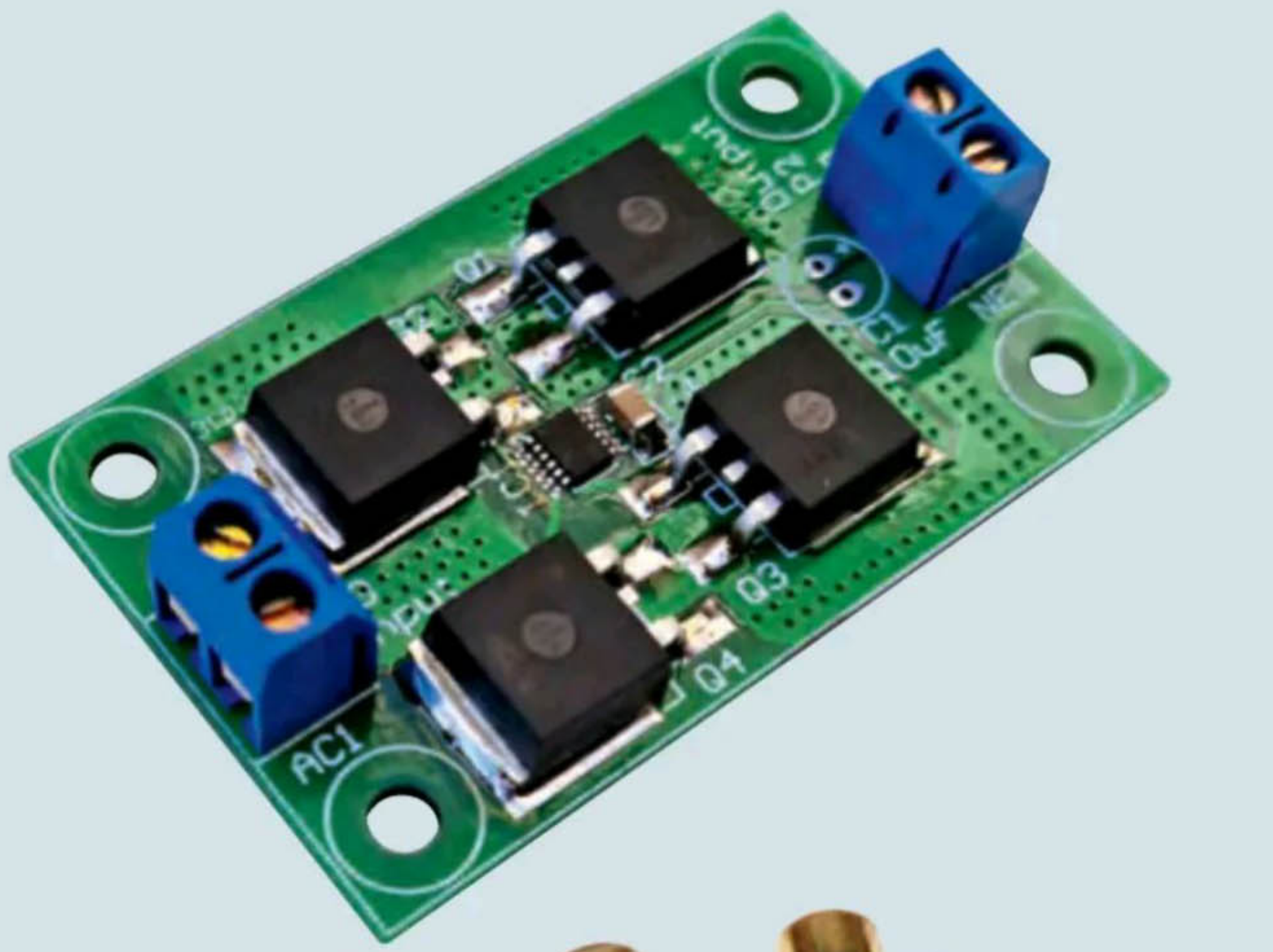
microchip.com/bluetooth

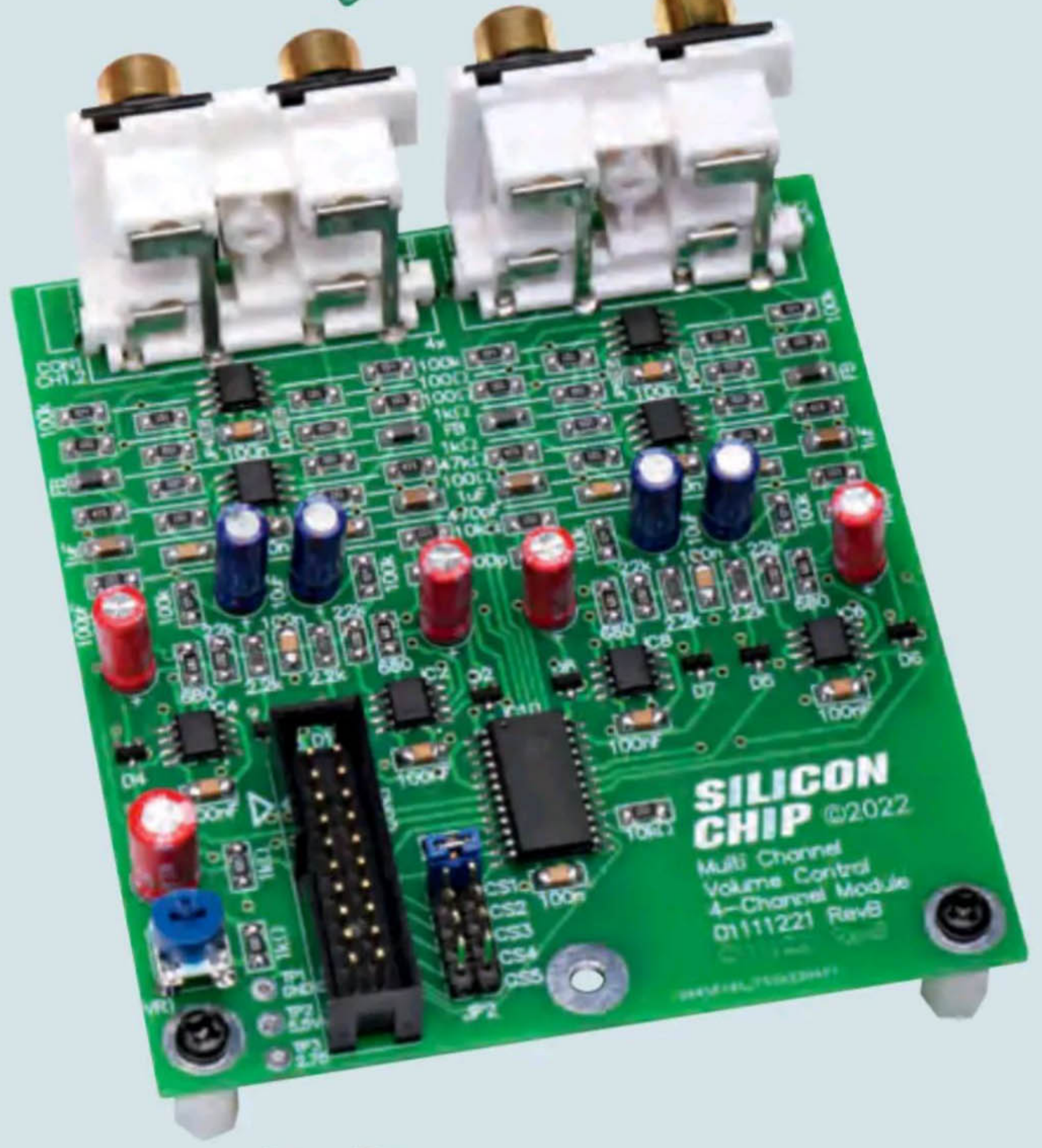
The Microchip name and logo and the Microchip logo are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. All other trademarks are the property of their registered owners.
© 2024 Microchip Technology Inc. All rights reserved.
MEC2578A-UK-07-24

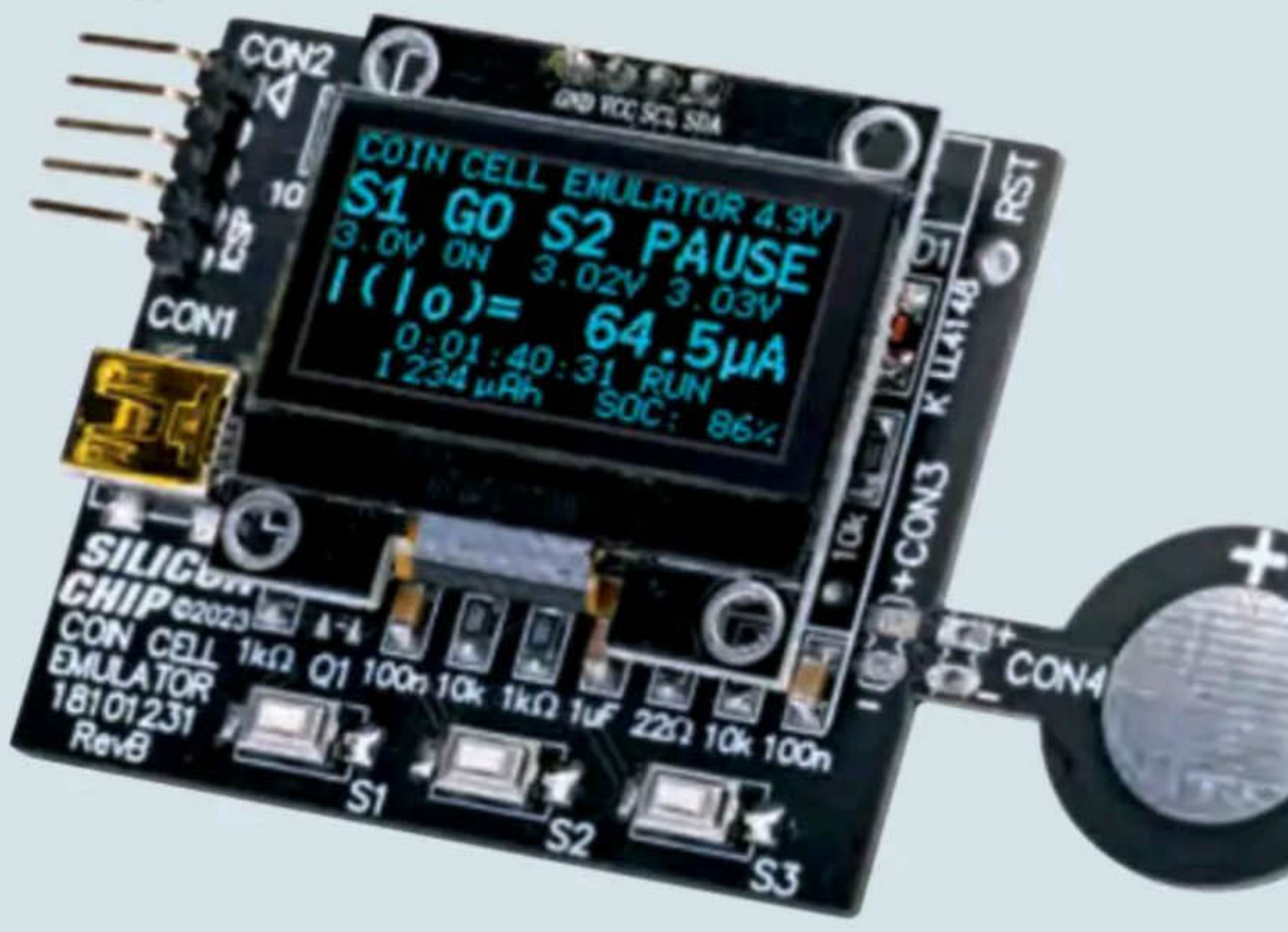
Practical Electronics

Volume 53. No. 11
November 2024
ISSN 2632 573X

Contents







Projects and Circuits

Ideal Diode Bridge Rectifiers by Phil Prosser 8
Active rectifiers have much greater efficiency than silicon diodes, and as a result, run much cooler. We show how to make up to six different Bridge Rectifiers depending on how much space you have and how much power you need to handle.

Multi-Channel Volume Control, part one by Tim Blythman 30
People with home theatre/surround sound systems need to control the volume of six or more audio channels. This Multi-Channel Volume Control has a touchscreen and receives infrared remote signals. It can be expanded to handle up to 20 channels.

Coin Cell Emulator by Tim Blythman 52
When designing or testing a device that runs from a coin cell, you need to know how much current it draws. That can be difficult given the low currents often involved. This device powers the circuit while showing the voltage, current and other helpful statistics.

Series, Features and Columns

The Fox Report by Barry Fox 3
Thermal batteries show considerable promise for domestic hot water

Net Work by Alan Winstanley 4
On this 60th anniversary of *Practical Electronics* magazine, Alan revisits the history of the magazine. In doing so, he touches on British contributions to today's Internet.

Practically Speaking by Jake Rothman 16
I recently had a shocking experience; here is some advice so that you don't!

Max's Cool Beans by Max The Magnificent 19
There's more than one way to blank zeros.

Teach-In 2024 – Learn electronics with the ESP32 by Mike Tooley 42
Part 9 – Putting it all together (the final part in this Teach-In series)

Techno Talk by Max The Magnificent 51
Robots and ukuleles

Circuit Surgery by Ian Bell 61
Digital signal processing – digital-to-analog converter (DAC) frequency response

MG4 XPower electric vehicle review by Julian Edgar 66
Julian has spent nine months and 12,500 miles (20,000km) behind the wheel of his MG4 XPower EV. He has found a lot to like about it, but it is not without its flaws.

14-segment, 4-digit LED Display Modules by Jim Rowe 72
Instead of seven segments, this LED display module has 14 per character, so it can display letters, digits and even a few symbols. It has a built-in I²C serial interface, allowing popular microcontrollers like the Arduino Uno or Nano to drive it easily.

Regulars and Services

Editorial 2
The challenges of making electronics work worldwide

Reader services – Editorial and Advertising Departments 2

PE Teach-In 9 collected articles book with CD-ROM 17

Practical Electronics back issue downloads 18

PE Teach-In 8 collected articles book with CD-ROM 29

Wireless for the Warrior 60

Subscribe to Practical Electronics and save money 71

Practical Electronics PCB Service 76
PCBs for Practical Electronics projects

Classified ads and Advertiser index 79

Next month! – highlights of our next issue of Practical Electronics 80

From the 1st of July 2024, subscriptions and renewals paid by PayPal, Visa and Mastercard are now handled via our website: www.electronpublishing.com

Click the Subscriptions link at the top of the page.

Select Publishing Services will continue to handle renewals via phone, cheque and mail.

Made in Australia & the UK.

Written in Britain, Australia, the USA and Ireland.

Read everywhere.

© Silicon Chip Publications Pty Ltd 2024

Copyright in all drawings, photographs, articles, technical designs, software and intellectual property published in *Practical Electronics* is fully protected, and reproduction or imitation in whole or in part are expressly forbidden.

The December 2024 issue of *Practical Electronics* will be published on Monday, 4th of November 2024 (see p80).

Practical Electronics

Volume 53. No. 11
November 2024
ISSN 2632 573X

Editorial

Editorial offices
Electron Publishing (Aust.) **Tel** +61 2 9939 3295
Unit 1, 234 Harbord Road **Email** pe@pemag.au
Brookvale NSW 2100 **Web** www.electronpublishing.com
Australia

Address mail to:
Electron Publishing (Australia)
PO Box 194, Matraville NSW 2036
Australia

Advertising enquiries
+61 2 9939 3295 pe@pemag.au

Editor Nicholas Vinen
Publisher Nicholas Vinen
Digital subscriptions Stewart Kearn **Tel** 01202 880299
Online Editor Alan Winstanley
Web Systems Kris Thain
Production Bao Smith
Technical staff Tim Blythman, John Clarke

Print subscriptions
Practical Electronics Subscriptions
PO Box 6337
Bournemouth BH1 9EH **Tel** 01202 087631
United Kingdom **Email** pesubs@selectps.com

Technical enquiries
We regret that technical enquiries cannot be answered over the telephone. We are unable to offer any advice on the use, purchase, repair or modification of commercial equipment or the incorporation or modification of designs published in the magazine.

Questions about articles or projects should be sent to the editor by email: pe@pemag.au

Projects and circuits
All reasonable precautions are taken to ensure that the advice and data given to readers is reliable. We cannot, however, guarantee it and we cannot accept legal responsibility for it.

Some projects and circuits published in *Practical Electronics* employ voltages that can be lethal. Do not build, test, modify or fix any mains-powered equipment unless you fully understand the safety aspects involved and you use an RCD (GFCI) adaptor.

Component supplies
Silicon Chip Publications may offer kits or other parts for making our projects, but not in all cases. When kits are not available, readers will need to find and source parts themselves. We advise readers to check that all parts are still available before commencing any project in a back-dated issue.

Advertisements
Although the proprietors and staff of *Practical Electronics* take reasonable precautions to protect the interests of readers by ensuring as far as practicable that advertisements are bona fide, the magazine and its publishers cannot give any undertakings in respect of statements or claims made by advertisers, whether these advertisements are printed as part of the magazine, or in inserts. The Publishers regret that under no circumstances will the magazine accept liability for non-receipt of goods ordered, or for late delivery, or for faults in manufacture.

Transmitters/bugs/telephone equipment
We advise readers that certain items of radio transmitting and telephone equipment which may be advertised in our pages cannot be legally used in the UK. Readers should check the law before buying any transmitting or telephone equipment, as a fine, confiscation of equipment and/or imprisonment can result from illegal use or ownership. The laws vary from country to country; readers should check local laws.

The challenges of making electronics work worldwide

One of the challenges we constantly face when developing electronic projects is making it so that people just about anywhere in the world can build them. We do most of the design work in Australia but they need to be suitable for readers in the UK and Europe (and also the USA) as most *PE* readers live in those areas.

Mains-powered projects are a particular challenge, with varying voltages, plugs, sockets and so on. Using IEC plugs/sockets and universal switch-mode supplies helps a lot but those things can get expensive, and sometimes they just aren't suited to a particular design.

Parts sourcing is also a challenge. We try to use 'generic' parts where possible, but sometimes a suitable one isn't available and it can be a real chore trying to find overseas suppliers that have compatible parts. Some suppliers even sell parts that are unique to them! At least many major suppliers like DigiKey, Mouser and Farnell will send parts to just about anywhere.

No Audio Out column this month

Jake Rothman was simply too busy to write his usual *Audio Out* column this month, although he did manage to pen a short *Practically Speaking* item. We expect normal service to resume as soon as his other commitments allow.

Some PCB orders were accidentally delayed

Apologies to readers who ordered PCBs during August and the first few days of September and didn't receive them until several weeks later. We had an error in our configuration that caused some orders not to be sent to the shop manager.

It wasn't until someone complained they had not received their PCB that we realised what was going on. It should be fixed now, and everyone who ordered a PCB or PCBs during that period should have them by now.

Contacting us

Part of the reason we didn't realise there was a problem with PCB orders is that some customers emailed Electron Publishing (UK) regarding orders; they are no longer directly involved in the magazine. We are not sure where those emails go but it seems to be a black hole. Apparently, those old addresses were still on the *PE* website for a while after we took over; they have now been changed.

For order enquiries, or other emails related to the magazine, please use our newly set up email address: pe@pemag.au

Nicholas Vinen, Electron Publishing (Australia)*
Publisher & Editor,
Practical Electronics Magazine

* a division of SILICON CHIP Publications Pty Ltd.

The Fox Report

Barry Fox's technology column

Turning up the heat on batteries

As lobbyists continually remind us, the problem with renewable energy is that it's unreliable.

Solar only delivers when the sun shines and wind only when it's windy. So, all eyes are on power storage. With the high prices and limited capacity of lithium batteries for solar storage, you will know what's not (yet) the answer.

Alternative solutions like pumping water up hills is only viable on an industrial scale. What's needed is an affordable consumer store. I'm surprised there has been so little talk of heat stores or heat batteries.

I first became interested when a friend in the country, with no online gas supply and facing ever-increasing prices for oil – which can be stolen from external tanks on dark nights – installed electric heating.

My country friend had already paid a huge price for EV solar power and big batteries but there was no way this could provide hot water. There was no space for large, insulated hot water tanks and no three-phase mains available.

But the installed system, with only a small water "tank", can run instant hot baths. How? The system maker, Fischer, provided a pile of loosely-related paper manuals as impenetrable as early computer tomes. They mainly tell how to program and control the radiators. There was very little on the big question: how is the water being so rapidly heated?

Fischer's web postings and sales publicity promise to tell "Everything About Our Heaters And What Makes Them Special". But the details on offer are very sketchy:

Aquafficient [the trade name for the Fischer water heater] is an innovative, modern and highly efficient solution to heating water... it uses a single heating element and patented thermal storage technology, to provide hot water without the need for tanks.

Instead, the hot water is produced from stored heat energy within a unique phase change material… known as a heat battery. Once the battery has been fully charged, it will pass the heat energy that has been stored inside to the cold water coming into the unit… the cold water comes into a highly efficient heat exchanger that has been embedded within the battery, and this transfer of heat only happens when hot water is called for.

This electric water heater provides hot water until the battery is depleted. The battery will recharge itself to ensure you have access to hot water 24/7. [It has a] Life span of 40,000 cycles, equivalent to 50 years of usage.

There is no detail given on the PCM (phase change material) used and who invented it. Out of curiosity, and because I suspect our readers will be interested in the under-discussed concept of heat storage, I decided to do some digging.

To cut to the chase, British company Sunamp (**https://sunamp.com/en-gb/**) developed, patented and manufactures a practical and reliable PCM heat battery technology that other companies now build into their heating systems.

Although I found no mention of Sunamp in Fischer's promotional material, and no mention of Fischer in Sunamp's publicity, the University of Edinburgh, where the key PCM research was carried out, names Fischer as a user of Sunamp technology. This is to be found in a detailed research Case Study published in 2021 by Edinburgh University's School of Chemistry, EaStCHEM: **https://pemag.au/link/abzb**

The nitty gritty of that lengthy report is that the idea of PCM storage is old, but was until recently commercially flawed.

The heart of a heat battery is phase change material (PCM) that absorbs heat when melting and releases it when it freezes. It has long been known that, in theory, salt hydrates make ideal PCMs because of their high energy density, plentiful supply, low cost, safety and sustainability.

In theory, sodium acetate trihydrate (SAT), with a melting point of 58°C, is an ideal PCM for domestic use. But the practical downside, which Sunamp found in 2006, was that SATs change their chemistry in use, mainly through the formation of solid anhydrous sodium acetate. So, like an electric battery, the heat battery degrades with use.

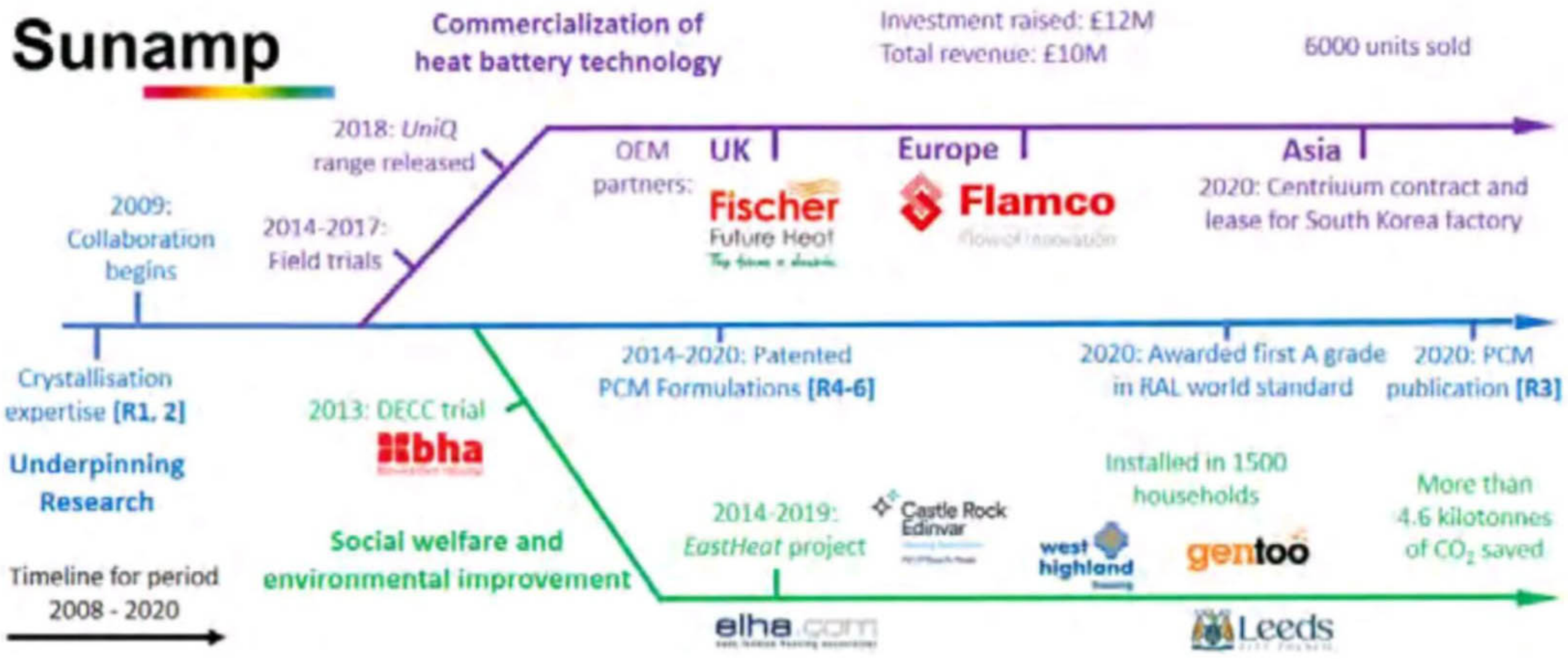
EaStCHEM Professor Colin Pulham, who specialised in crystallisation science, helped Sunamp crack the problem by adding polymers. It then remained chemically stable for up to 40,000 heating and cooling cycles. Sunamp extrapolates this to a predicted lifetime of over 50 years, as Fischer claims in its sales literature.

The University report cites Fischer Future Heat as a key partner and quotes Fischer as describing the technology as a "must have" hot water solution. A big plus is the option to heat the PCM with off-peak electricity – or solar power – and then use the heat to run hot water later.

Gluttons for technical and chemical detail can go to the three source patents, all readily available to read free on line.

The original patent (**WO/2014/195691**, Improved Phase Change Compositions) was filed in June 2013 by Sunamp Ltd and three inventors: Andrew Bissell, David Oliver and Colin Pulham, Professor of High-Pressure Chemistry at the School of Chemistry in Edinburgh University.

Two more patents on improved PCMs followed, also naming Professor Colin Putnam: **WO/2015/025175** (August 2013) described the use of Strontium Bromide and **WO/2020/074883** (October 2018) with metal nitrates added to the mix. **PE**


Sunamp
Commercialization of heat battery technology
Investment raised: £12M
Total revenue: £10M
6000 units sold
2018: UniQ range released
OEM partners:
UK
Fischer Future Heat
Europe
Flamco
Asia
2020: Centriuum contract and lease for South Korea factory
2009: Collaboration begins
2014-2017: Field trials
Crystallisation expertise [R1, 2]
Underpinning Research
2014-2020: Patented PCM Formulations [R4-6]
2020: Awarded first A grade in RAL world standard
2020: PCM publication [R3]
2013: DECC trial
bha
Social welfare and environmental improvement
2014-2019: EastHeat project
Castle Rock Edinvar
west highland
Installed in 1500 households
gentoo
More than 4.6 kilotonnes of CO₂ saved
elha.com
Leeds
Timeline for period 2008 - 2020

Net Work

Alan Winstanley

This month we celebrate 60 years of Practical Electronics and recount the tale of how the magazine came about. Net Work also gives an account of Britain's often overlooked influence when the Internet's foundations were first taking shape.

A decade ago, I wrote a two-part piece celebrating fifty years of *Practical Electronics*, Britain's first hobbyist magazine dedicated to exploiting transistor and integrated circuits. This month's edition sees us reaching our diamond jubilee, as the November 2024 issue marks sixty years of successful publication.

The November 1964 issue was released in an era that embraced the race to the Moon; many developments in modern electronics owe their origins to the 1960s space program.

The roots of *Practical Electronics* date much further back, though, to an age when semiconductors were inconceivable and fabricating a DIY 'wireless' meant scratch-building a radio receiver using high-voltage valves (vacuum tubes), eagerly following the skeleton plans printed in newsletters and magazines of the time.

At the turn of last century, the impact of radio communications was as momentous as the Internet, email and social media have been in today's digital age. Radio constructors were eager to 'tune in' to the new medium and they relished the challenge of using their practical and theoretical skills to build radio receivers and aerial arrays of every description.

Primitive earpieces hooked to home-made receivers buzzed and crackled to the sound of radio transmissions that would eventually be broadcast from all around the world.

FREE INSIDE—24 PAGE WIRELESS BOOK!
HOME CRAFTS
WOOD WORKING
MODEL MAKING
AMATEUR MECHANICS
ETC. ETC.
Hobbies
AND
PRACTICAL WIRELESS
2D
Make this
MIDGET ONE VALVER
FULL INSTRUCTIONS AND WIRING DIAGRAM IN GIFT BOOK

Hobbies and Practical Wireless magazine, published in 1932 by George Newnes Ltd.

The earliest radio magazine I'm aware of first appeared over 110 years ago. *The Marconigraph* of April 1911 was published in London by none other than Marconi Wireless themselves. They marked the new era of radio with these opening lines:

"Seventeen years ago [in 1894] the idea of wireless telegraphy for the transmission of wireless messages through space was regarded as an utter impossibility, yet within a year of that date the 'impossible' was achieved, the outcome of which is the present day world-wide system of wireless communication by land and sea."

The Marconigraph aimed to acquaint the 'lay reader' with the possibilities of "this most marvellous new invention" and the journal vividly illustrated the explosive growth in radio transmissions being witnessed in every corner of the world. Early experiments with aircraft-based radio reception were also described. Marconi's magazine became the respected title *Wireless World* a few years later.

A multitude of magazines

With fascination in radio broadcasting intensifying, there soon appeared a multitude of magazines for radio constructors to choose from, such as *Popular Wireless*, a weekly title for listeners and hobbyists that in 1922 promised readers "the world's latest hobby fully explained".

Many titles appeared weekly, but as post-war paper shortages severely inhibited the production of magazines and newspapers, some were forced to adopt a monthly timetable instead. Other titles included *Radio Constructor*, which became *Radio & Electronics Constructor* that many of us saw jostling for position on bustling newsstands up until the 1980s.

Editor's note: the first issue of Wireless Weekly was published in Australia on the 4th of August 1922, eventually leading to Silicon Chip.

One London-based publisher, George Newnes Ltd, started printing a literary newsletter in the late 19th century that was a precursor to journalism and some of the earliest British newspaper titles.

In my collection are some original *Hobbies and Practical Wireless* magazines from 1932, published by Geo. Newnes, which contained a very broad array of projects catering for wholesome DIY hobbies, including woodwork and fretwork, stamp collecting, model-making and German-made 'Trix' – an early form of Meccano that for budding 'home engineers' was a sensation at the time.

This early hobby magazine, printed on newsprint in three spot-colours, included skeleton plans for multi-valve wireless receivers, three-valve circuits being the most popular. An astonishing range of every conceivable radio component was sold by a thriving mail order industry.

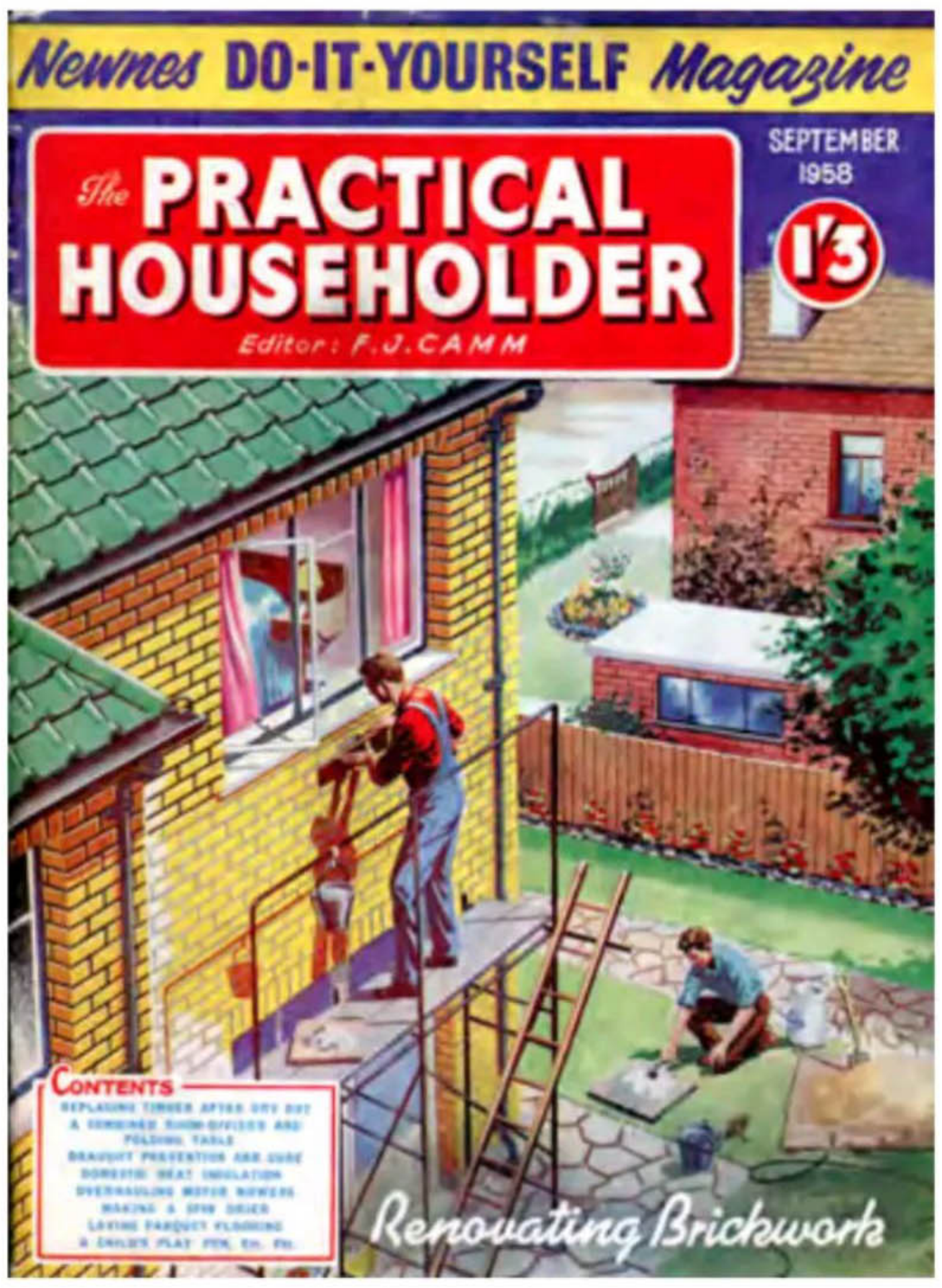
Just to complete the picture, amusing 'small ads' in H&PW featured freely-available air pistols and fearsome-looking catapults, exclaiming that "No boy is a boy without these!" Shiny new bicycles were advertised at a cost of £3 15s as well.

The rise of Camm's 'Practicals'

In the 1930s, a Newnes editor, the now legendary F.J. Camm, started a range of 'Practical' magazine titles aimed at the many hands-on, sleeves-up home mechanics, motorists and wireless enthusiasts who took such pursuits in their stride.

Practical Wireless would become a stand-alone title, and F.J. Camm was sympathetic about catering for those entering the world of hobby wireless construction for the first time. In the launch issue of September 24th 1932, he promised:

"Simplicity of Treatment AND, of great importance, particular care will be taken in presenting the contents in clear and simple language. Highly technical terms will be dispensed with wherever simple description can be employed, and diagrams will be prepared and explained so as to be readily understood. Thus the reader with a modest technical



Newnes' 'Practicals' magazines captured the post-war zeitgeist of British householders, hobbyists and motorists. Mixing mortar and climbing scaffolding was no problem for enthusiastic DIYers!

knowledge, or even the keen amateur, will find PRACTICAL WIRELESS appeals to him as well as to the reader with a sound technical knowledge."

Such re-assurance was at odds with Camm's views of Do-It-Yourself enthusiasts, though. Some years later, at a time when some householders could tackle even quite ambitious homebuilding and repair projects for themselves – including building an entire house! – a novice wrote to a newspaper lamenting the lack of guidance that was available to budding DIYers.

He hoped that projects and articles might expand more on the fundamental skills needed to complete basic tasks. In a brutal July 1958 Editorial in another 'Practical' title that he edited, The Practical Householder, Camm branded those without the practical skills as "simpletons", adding that "for those who are mentally incapable of learning, nothing can be done. There are dunderheads in all walks of life." Charming!

Practical Wireless became a very familiar sight on British newsstands; sales reached 120,000 copies a month as interest in radio technology continued unabated.

However, another branch of space-age technology was emerging that offered 'solid-state' solutions for many other electronic applications, not just radio. The transistor had arrived on the electronics scene. New circuit techniques were fast being developed and more magazine pages were also needed to cater for an abundance of advertisers, such was the hobbyist's appetite for constructing electronic projects using these new semiconductor devices.

Toying with the idea of homing in on this new market, George Newnes Ltd. considered creating a new magazine with a working title of *Practical Electronics and Hi Fi*. In 1962, Newnes recruited ex-RAF wireless operator and former EMI technical publications staff member Fred Bennett, who duly started work at Newnes supposedly as a sub-editor on *Practical Wireless*, learning how the publishing industry worked.

Fred wondered to himself whether his role was really necessary, though. "Even I could see that there was hardly any need for additional staff", he reflected many years later. Was something brewing on the electronics magazine front?

Pondering Practical Electronics

Later that year, Managing Editor Bert Collins, the successor to F.J. Camm (who died in 1959 aged just 63), offered Fred the opportunity to work on a new magazine that would complement *Practical Wireless*.

The new title had been nailed down simply as *Practical Electronics*, although a US magazine of the same name was first published five years earlier, in 1957. Published by Fawcett of New York, it offered designs for a pocket-size transistor radio and transistor checker, amongst other projects.

It's likely that this new venture was why Fred had been recruited by Newnes all along. Fred jumped at the chance, and set about designing a pilot issue of *Practical Electronics* from scratch, while still working on the *PW* desk.

Fred described how he then spent a lengthy and anxious year waiting for feedback about his proposals when, suddenly, at the end of 1963, Bert Collins gave him the green light. An all-new *Practical Electronics* would launch before the end of the following year.

With a new office location to accommodate their team, *PE* would be independent of its wireless forebear. The first November 1964 issue of *Practical Electronics* was a success, with 115,000 copies being sold.

Reflecting on the matter many years later, Fred had nothing but praise for the management at George Newnes Ltd, whose foresight recognised the potential of newly-emerging applied semiconductor electronics.

The magazine set the pace in Britain for producing advanced, professionally-presented and sometimes highly complex hobby electronic projects. It was largely unchallenged for many years.

Wikipedia has more details on George Newnes Ltd at **https://w.wiki/B3pS**

With publication of *Practical Electronics* in full swing in a very vibrant market, Fred realised that they had "begun to tap an inexhaustible source of talent... It was as though electronics enthusiasts had been anticipating our appearance."

Such was the embarrassment of riches submitted by highly skilled and immensely talented electronics engineers and professionals of the time that the magazine never faced a shortage of material. This was also true of the readers' 'circuit ideas' column called Ingenuity Unlimited, which first appeared in May 1965 and was soon heavily over-subscribed. It inspired me to submit my own little article in the mid 1970s, earning me £5 pocket money!

PE also rallied to support novices or less gifted readers. Fred Bennett was keen to engage with beginners in electronics and encourage them to learn and broaden their skills and knowledge. There was plenty to challenge the hobbyist, with a wide variety of digital and analog projects, tutorials, news briefs and those all-important mail-order adverts packed into every issue.

In 1968, Fred offered a young Mike Kenward a job as a technical sub-editor. Mike would eventually assume the role of Editor and publisher many years later, establishing Wimborne Publishing Ltd for the purpose. The October 1974 edition celebrated the magazine's first decade, acknowledging that the integrated circuit had since become "universally popular" with constructors pursuing their hobby.

An Everyday experience

Fred developed a sister magazine title in 1971 – *Everyday Electronics* – written by the same team. It was aimed at beginners and younger readers looking to build

Practical Electronics
NOVEMBER 1964
PRICE 2'6
EXTRA! ELECTRONIC DATA BOOKLET
plus
FREE
INSIDE
COLOUR CODE CALCULATOR
for
RESISTORS & CAPACITORS
TRANSISTORISED EQUIPMENT
INTEGRATED AUDIO AMPLIFIER
GEIGER-MULLER RATEMETER

The first edition of *Practical Electronics* in November 1964 catered for constructors seeking to build a wide range of semiconductor-based projects.

everyday electronics
NOV. 71
15p
A NEW MAGAZINE BRINGING A FASCINATING HOBBY TO EVERYONE
PROJECTS EASY TO CONSTRUCT
THEORY SIMPLY EXPLAINED
FREE IN THE No1! ISSUE
PRINTED WIRING BOARD
RECORD PLAYER
FREE WITH NO. 1 OF everyday electronics
PRINTED WIRING BOARD

Everyday Electronics was designed to offer simpler projects and tutorials for beginners and younger hobbyists.

less-challenging electronic circuits and gadgets. Coincidentally, the June 1967 issue of *Radio-Electronics*, published in the USA by Gernsback, was a special edition showcasing 'Everyday Electronics'.

'Teach-In' tutorials appeared in *EE* right from the outset as well, and were instrumental in helping readers to develop and expand their skills and knowledge. As a testimony to Fred's vision, we still publish a dedicated Teach-In series to this very day.

On a personal note, I once had the pleasure of speaking with Fred Bennett. I owe my entire electronics publishing career to the support and generous encouragement that he gave me during my teenage hobbyist years. You can see my efforts in PDF format at **https://pemag.au/link/ac0z**

Fred passed away in 2006, aged 85. A tribute written by Mike Kenward appeared on our legacy website, which is still online, at **http://www.epemag.com/vault/0306.htm**

Both titles changed hands and went their separate ways for a while, then merging into *Everyday Practical Electronics* (*EPE*) before finally settling down as *PE* once again when Editor Matt Pulzer acquired the title.

This year, *Practical Electronics* opened a new chapter in its story when Australia's Silicon Chip took over at the wheel; it's a perfect fit as *PE* had been publishing Silicon Chip projects since January 2006 after our home-grown capacity to source, edit, publish and technically support hobbyist projects had finally run its course.

I previously described in more detail the often tumultuous journey of the magazines in my 50 Years celebration in 2014, which can still be downloaded as PDFs from **https://www.epemag.com/resources.html**

Additional resources describing the history of, and people behind, both titles were written by former Editor Mike Kenward and currently remain online at our legacy website; see **https://www.epemag.com/epe-history.html**

I suppose I'm memorialising the story of this magazine for the benefit of future readers and contributors, in case these finer details are eventually lost in time. With sixty years of publishing now behind us, and with dedicated and committed staff at Silicon Chip now firmly established at the helm, there's plenty for the *Practical Electronics* reader to look forward to. Modern electronics has become accessible in a way that Marconi never envisaged more than a century ago. Hopefully it's not tempting fate to say that I'll look forward to seeing our Platinum anniversary in 2034!

The Internet's early days

A one-page Net Work column first appeared in the August 1996 issue and a URL appeared on the cover shortly afterwards. Now seems a good time to re-visit the roots of the technology that gave rise to this column to begin with.

In July 1996, when online search and reference resources were very thin on the ground, I wrote a feature in *Everyday Practical Electronics* titled "The Internet – What's in it for you?"

That article spelled out the story of the Internet thus far. You can still download a PDF scan for free from **https://pemag.au/Shop/3/7367**

No-one could know, however, how the Internet would ultimately dominate every corner of our society twenty or thirty years later.

Much has been documented about the creation of the Internet itself, so I won't repeat all the details again. A book entitled "Inventing the Internet" by Janet Abbate (ISBN 0-262-01172-7, 1999, MIT Press, 264pp) is a rare and thoroughly researched but highly readable academic work. It offers a detailed and granular account describing the often turbulent development of the Internet.

It's one of the best reference works that I've come across and I'll draw from some of its material for the remainder of this month's article. Moreover, it recounts the often overlooked contributions that Britain's unsung scientists and developers made when the Internet's foundations were being laid.

The launch of Sputnik by the USSR in 1957 galvanised America into developing a 'survivable' communications network for military use. The US non-profit RAND Corporation (short for Research & Development) benefited from receiving open-ended funding for researching military projects like these, so money was no object.

Credit goes to a young RAND engineer, Paul Baran, for designing an early form of distributed communications or 'message switching' system for the military that could withstand an enemy hit. There is more background to this at **https://www.rand.org/about/history/baran.html**

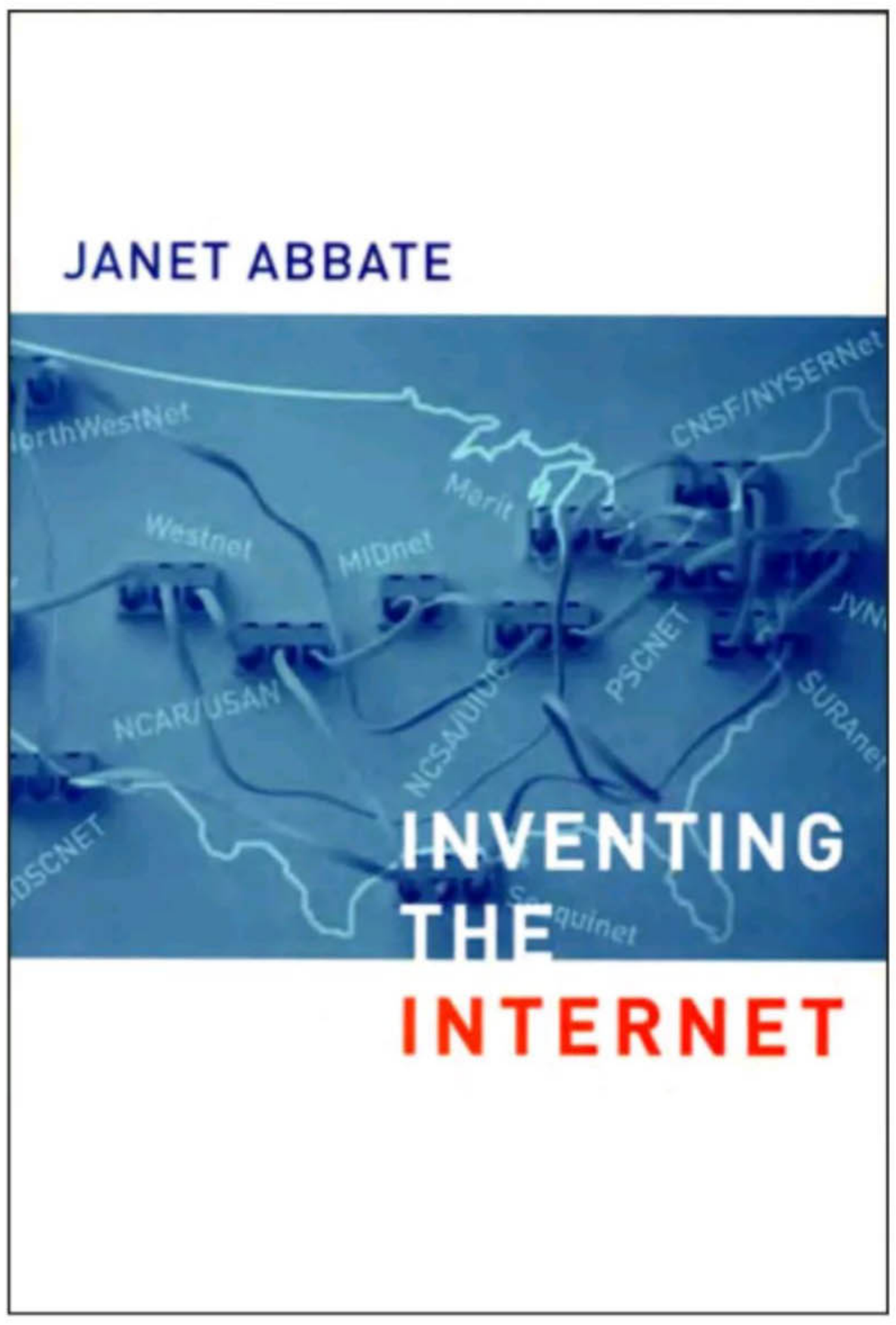
According to Janet Abbate, Baran had no confidence in the contractors' ability to build it and so, rather than risk being discredited, he scrapped the proposals. Nevertheless, his ideas were widely published and would become very influential in the coming years.

However, US scientists and academics didn't understand the need for 'survivability' at the time, so they weren't interested in resilient, bomb-proof (literally) message switching. Thus, the concept went largely undeveloped.

The white heat of technology

Here in Britain, Harold Wilson's Labour government championed the drive for new technologies and industries. His famous 'white heat of technology' conference speech in 1963 hoped to inspire the country into embracing a scientific revolution, dispensing with old restrictive practices that hampered the country's economic growth.

The British Library website hosts a recording of the speech at **https://pemag.au/link/ac10**



"Inventing the Internet" by Janet Abbate offers a detailed insight into the Internet's evolution, highlighting the influence of the NPL and is an easy-enough read.

Janet Abbate makes the point that the critical difference in Wilson's approach, compared with the USA, was that Britain under Labour would fund new technology to stimulate commercial and industrial production and boost the economy, rather than use it for military applications.

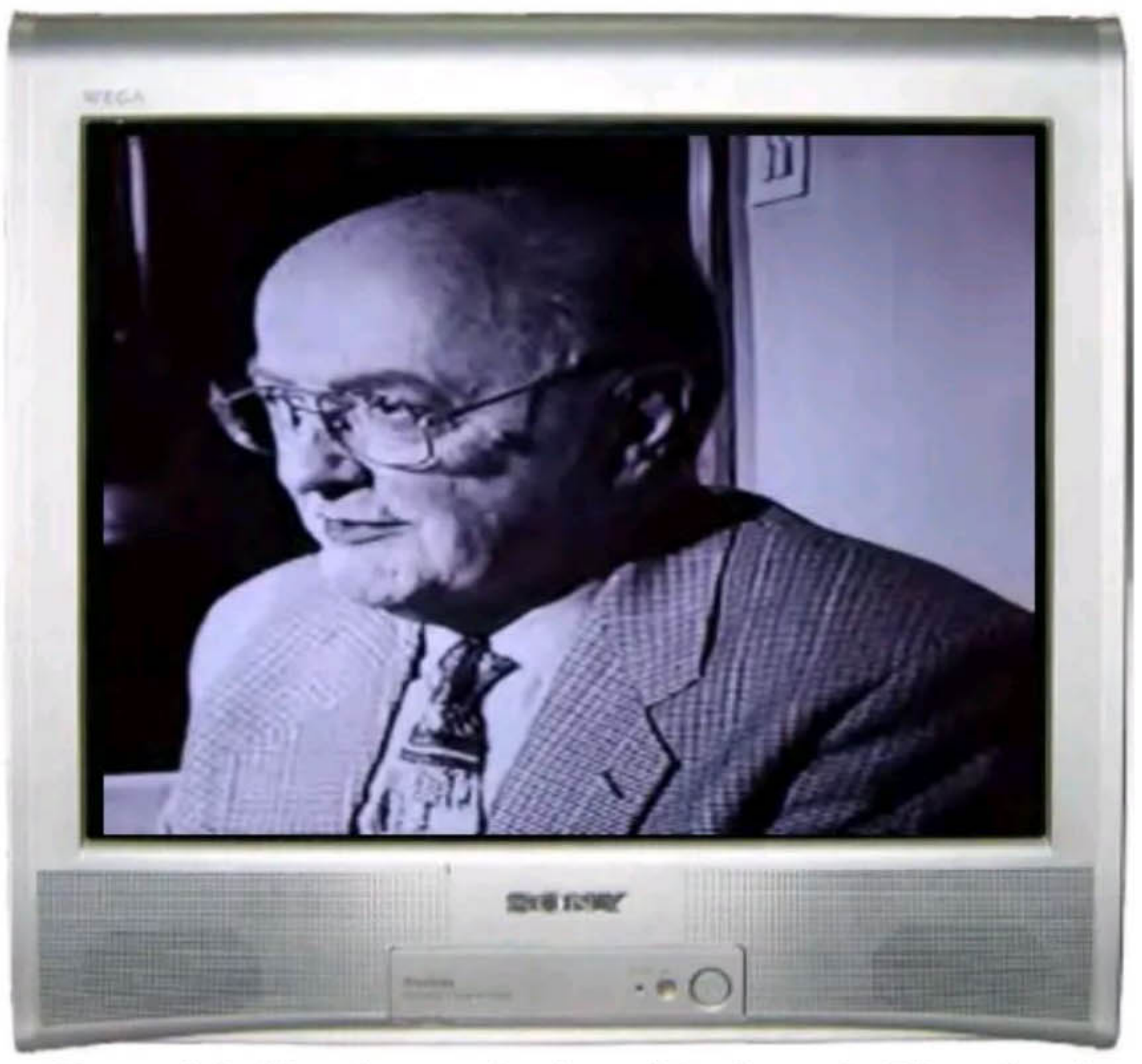
In contrast, in 1960s America, key university centres of technological excellence were created and given an open-ended design brief. They were tasked with investigating and solving problems in building the nation's defence systems.

It was emphasised that there would be no limitations to the research and there would be no project-specificity either. Doubtless, this approach encouraging 'blue sky thinking' would bring out the best of their students and academics. Ultimately, it gave rise to the ARPANET computer communications network.

It would be unfair to contrast this with Britain's own research programmes. For starters, the US government's near-limitless funding dwarfed anything available in post-war Britain. Secondly, the US approach was that system designers were building their own network for their own use, but Britain was heading in a different direction, focussing on developing user-friendly computer systems for commercial use instead.

A major part of Britain's computer research was carried out by the National Physical Laboratory (NPL), which had something of a shoestring budget.

One celebrated NPL computing engineer from Wales, Donald Davies, was instrumental in devising a new method of transmitting digital data that did not clog up and slow expensive telephone lines. He was working on the concept of splitting messages into smaller parts, to transmit data over long distances via a distributed network. Davies is credited with coining the term 'packet switching'.



Donald Davies of the National Physical Laboratory describes the concept of packet switching in a 1998 TV interview for the Open University (from BBC TV).

Unknown to him at the time, another form of 'packet switching' had previously been developed in the USA by Paul Baran, but the military application was not in the public domain.

Expensive timeshares

In an era of very expensive mainframe computers, a system of 'batch processing' of data was used to keep them continually operating. The concept of 'timesharing' was then devised to spread a computer's workload between multiple users. This enabled computers to run several programs in a multiplexing fashion, making the best of expensive computer time.

Davies, still working on a very tight NPL budget, learned of timesharing during a visit to the USA in 1965. He saw it as a way of maximising costly computer resources rather than being a 'survivable' datacomms system.

In 1966, ARPANET was just the germ of an idea and it seems the developers were originally unaware of Paul Baran's pioneering work. In 1967, NPL's Roger Scantlebury presented a paper in the USA describing their work on packet switching, and they explained NPL's ongoing research into building its own network, called the Mark I.

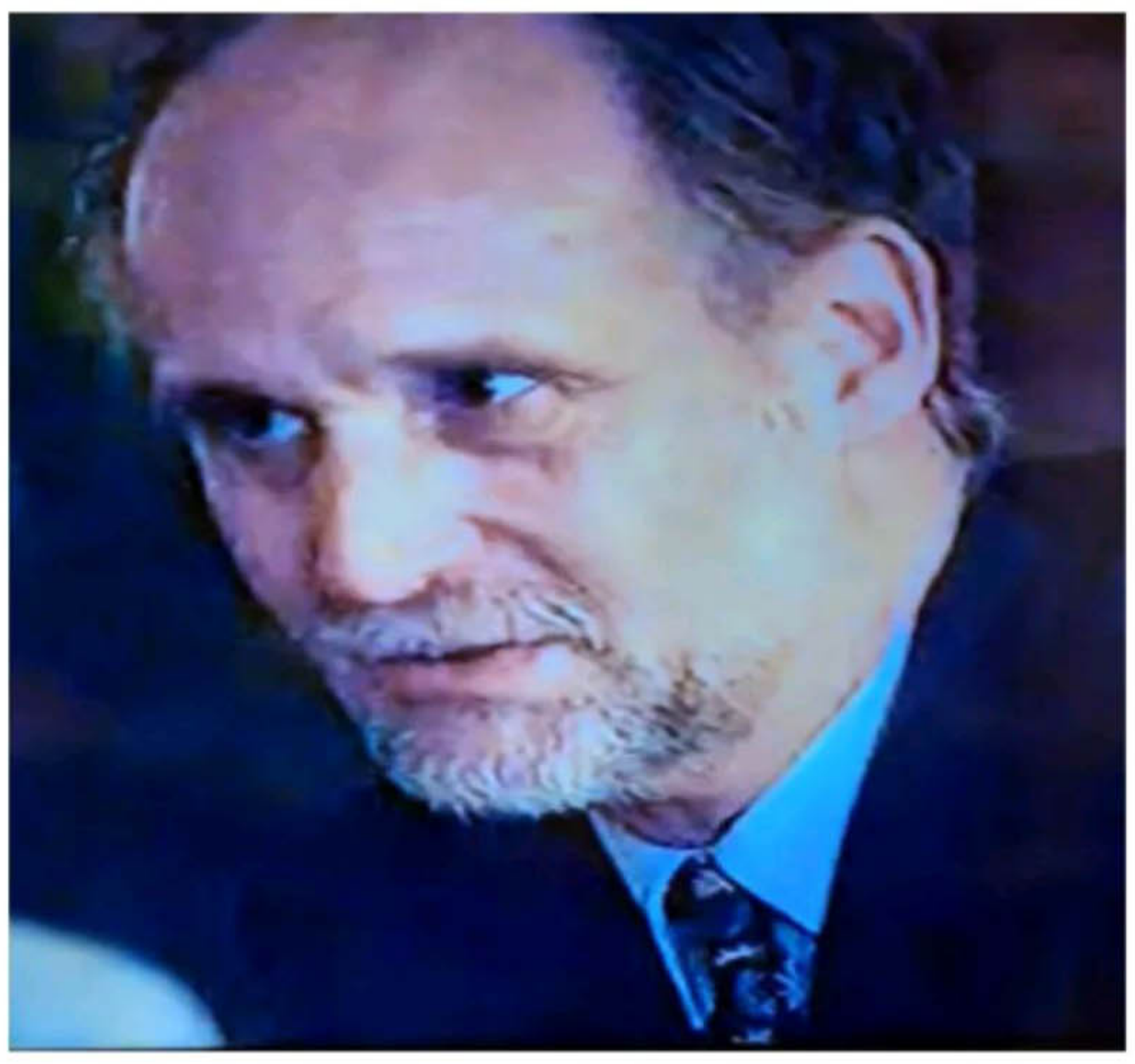
It is clear that NPL's own research influenced the design of the American ARPANET. Furthermore, British NPL scientists are credited with 'joining the dots' that brought RAND's original work and ARPANET's developers together to create a resilient packet-switching network.

Donald Davies is also recognised as the proponent of a dedicated network interface computer. As early network trials commenced in the USA, the first 'router' (a Honeywell IMP – Interface Message Processor) was developed. Built like a tank, the wardrobe-size 'peripheral' needed a helicopter and forklift truck to install it at UCLA on time.

Donald Davies also convinced the Americans that network bandwidth would be the key to everything so, according to Abbate, ARPANET's Larry Roberts duly upped the network spec. to a dizzying 56 kilobits per second, the maximum possible over analog phone lines (as every dial-up modem user of the 1990s would later discover).

A scaled-down network

Back home in Britain, Donald Davies remained keen to develop a network timesharing client that anybody could use, but he struggled with meagre budgets and a lack of resources. Furthermore, the General Post Office (GPO), which operated Britain's analog phone system at the time, wasn't interested in digital data communications anyway.



Roger Scantlebury of the NPL explained their work on packet switching to an American audience (BBC/Open University).

The British government dropped the ball. NPL proceeded to build Mark I, its own scaled-down network, using a Honeywell 516 computer. Very unusually, and showing great foresight, system resources were attached to the network where they could be accessed by network clients, rather than being hosted by isolated client computers. An upgraded Mark II network followed which, astonishingly, ran from 1973 to 1986.

Eventually, the GPO relented and after four years launched a packet-switching network in 1977. However, its next-generation system would use American hardware.

As the American network continued to grow, otherwise-incompatible University sites connected together using the new common protocol of TCP/IP (transmission control protocol/internet protocol). A mobile data experiment based in Hawaii proved successful, and a satellite link with Norway followed, initially exchanging seismic data.

The University College London connected with Norway shortly afterwards. In the late 1980s, the ARPANET was finally pensioned off when control of its infrastructure was transferred to the National Science Foundation Network (NSFNET). The Internet would then be 'demilitarised' and commercialised.

Much of the outstanding work done at the National Physical Laboratory by Donald Davies and his team was technically ahead of its time but, lacking funds, it could never compete with America's fully-funded military ARPANET.

Davies died in 2000, and we'll never know what the 'net might have looked like if the National Physical Laboratory had managed to design a messaging system on a much grander scale. At least Britain's hand was on the tiller when the idea of packet switching networks was first floated nearly sixty years ago.

As always, you can email the author at alan@epemag.net PE

POS
NEG
AC1
AC2
C2
Q1
Q3
IC1
J1
J2

Ideal Diode Bridge Rectifiers

By Phil Prosser

Rectifiers have evolved a lot over the last century, from selenium piles and mercury arc rectifiers to vacuum tube diodes, then germanium and silicon diodes. Now, active rectifiers offer much greater efficiency than silicon diodes, running much cooler. We show you how to make up to six different Bridge Rectifiers depending on how much power you want.

In the simplest terms, an 'ideal diode' uses a power Mosfet with a control circuit to replace a rectifier diode. Combining four such devices gives you an 'ideal bridge rectifier'.

While they are not truly ideal, they are much closer than a regular diode, with a forward voltage (and thus power loss and heat dissipation) typically around ⅒ that of a normal diode.

This idea caught my attention because I realised it would allow us to build devices like power amplifiers or power supplies that operate more efficiently and deliver more power, as less is lost in the bridge.

Bridge rectifiers used in large power amplifiers need a lot of heatsinking! They can dissipate tens of watts under heavy load.

That all changes with this design, which is a drop-in replacement for many existing bridge rectifiers.

When designing my Dual Hybrid Power Supply (to be published in an upcoming issue), I wished I had the time to delve into these active bridges, as the power loss in a high-current DC power supply bridge is also significant. For example:

- The PB1004 10A bridge rectifier has a forward voltage drop of over 1V at 5A, or 2V across the bridge. This means it is dissipating 10W at 5A.
- The KBPC3510 35A bridge rectifier drops 1V at 10A, resulting in a 2V loss and 20W dissipation at 10A.

The 2V drop is manageable, if annoying, by increasing the transformer voltage. However, transformers often come in 5V steps, meaning you might be wasting a lot of power to compensate for that relatively small voltage loss.

On the other hand, that 10-20W dissipation is troublesome, as it demands a substantial heatsink and forces physical layout decisions to enable this heat to be dissipated.

Lessons learned during the design process

The design of these modules served as a reminder on the need for attention to detail and the value of peer review. I did the bulk of the PCB layout while I was on holidays, and since there were only seven parts, what could go wrong? Plenty.

When I was making the CAD library for the LT4320 IC, I stuck the 'pin' that denotes the thermal pad for the IC in the wrong spot. This led me to assume it connected to the positive pin rather than the negative, where it belonged. I then laid out seven variants of this board from the schematic, all with the pad connected to the wrong output.

I now know that the LT4320 will work for several minutes with the thermal pad tied to the wrong pin, but after that, it will blow up, take out your Mosfets and short your transformer! I found the bug after blowing many fuses, £50 worth of bits, wasting a couple of days, and my whole budget of four-letter words.

To add insult to injury, I had to respin all the different prototype boards, another £50 lesson. Ouch! All for about 2mm of misplaced PCB trace.

Pros and cons

By comparison, if we use an LT4320 'ideal bridge' controller and TK6R9P08QM power Mosfets, we will see 70mV maximum drop per device at 10A, which is a total of 1.4W or about ⅒th of the heat you get from a standard bridge rectifier!

So what is the catch, and why aren't these used everywhere? I suspect there are a few reasons:

1. One of the complications that needs to be dealt with is generating the V_{GS} drive for the N-channel Mosfet, which requires a boost circuit to drive the gates well above the source voltages.

2. For a bridge, you need four power Mosfets and a controller, which increases parts count and cost.

3. The real benefits are accrued when rectifying lower voltages at high currents or if you cannot afford losses in your system (or when high efficiency is essential).

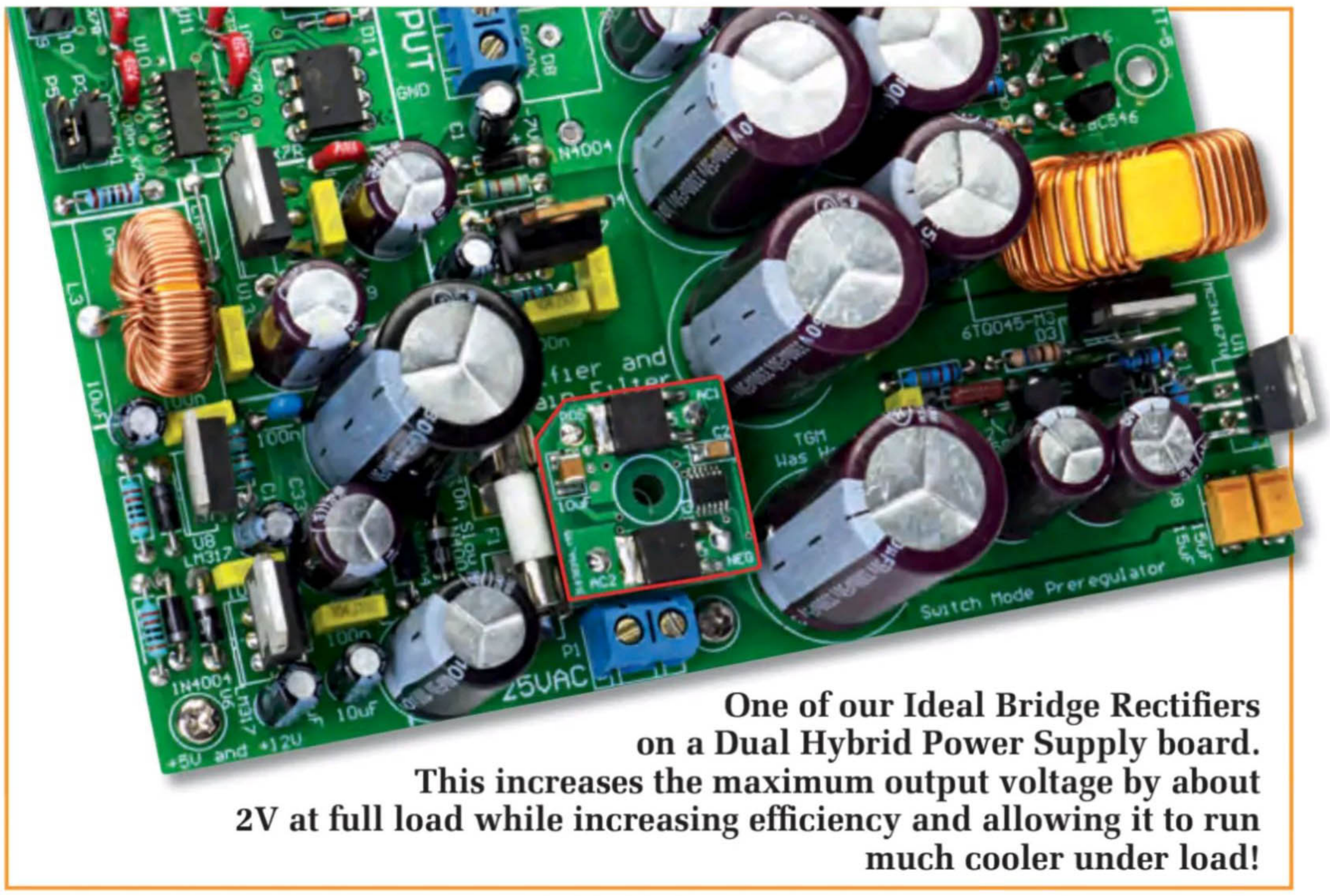
4. Because of how the control and switching works, for the simplest off the shelf solution, a dual-rail power supply (such as for a power amplifier) needs two bridges, each fed by one of the two secondary windings.

5. Your rectified output voltage rail needs to stay above 9V, or bad things happen (more on that later).

The best use cases for an ideal diode bridge rectifier are where space and capacity to dissipate power are limited, where voltage drop from the transformer is undesirable and where lower voltages at higher currents need to be rectified.

In terms of using Mosfets to replace

Ideal Bridge Rectifiers



One of our Ideal Bridge Rectifiers on a Dual Hybrid Power Supply board. This increases the maximum output voltage by about 2V at full load while increasing efficiency and allowing it to run much cooler under load!

diodes, it is interesting to note the growing use of 'synchronous' switch-mode converters.

In synchronous converters, the usual schottky diodes are replaced with power Mosfets. Many synchronous switch-mode controllers include an output to drive the diode replacement Mosfets, resulting in increased efficiency.

Design approach

Given the desire to investigate this technology, our efforts turned to an integrated solution. We wanted an option that could be used in a range of projects and showcase the potential of this technology, without making construction too tricky or the device too expensive.

A survey of ideal diode controller ICs shows that many are intended for hot-swap and redundant power supply applications. In this case, multiple power supplies are combined in an 'OR' function so that if one supply fails, the other picks up the load. Supply currents can be very high in a server application, so reducing diode losses is critical.

We also found several controllers for automotive applications, in alternators and circuit protection. These are generally intended for single-rail applications and are not suited to more general AC rectification. In particular, most utilise the diode to operate the circuit itself. This limits their application as generic diode replacements.

The range of available parts in this field is growing, so new ICs that are useful in a range of applications are coming on the market. In this project, we show how to use the most available controller IC and build a range of 'ideal diode bridge rectifiers' that can replace conventional diode bridges in various projects.

The controller we have selected is the LT4320, as this allows simple and compact boards to be built, ranging from tiny SOT-23 Mosfet based bridges through DPAK (TO-252) to very high current TO-220 based through-hole versions. Where might each of these be used?

- The SOT-23 Mosfet based bridge PCB is only 9 × 15mm and can be used inline on the DC power supply lead to a device or soldered in place of a small bridge. This can make the power lead for your device polarity agnostic without affecting its operation noticeably.
- Our boards using DPAK SMD Mosfets can replace the common 5mm pitch 19mm SIL bridge or rectangular bridges with corner pins or spade connectors (see the photo above) and handle high currents.
- There are also two 'standalone' versions that are basically just small boards you can mount in a chassis to provide the rectification function. One uses TO-220 Mosfets and other through-hole parts and can handle very high currents, limited mainly by the PCB itself!

There are a few limitations or requirements we need to work with that initially may sound onerous. However, in a real-world application, the following are not that hard to meet:

- The LT4320 works in a 'single-rail' configuration only.
- For an audio amplifier, you need to rectify the outputs of the two secondary windings independently. You then connect the negative output from one bridge to the positive output from the second bridge to get your split supply, usually at the main capacitor bank.
- We have achieved pin compatibility for all the larger bridge types. But DIP-8 and W02/W04 type bridges are a bit small for us to match, so if replacing one of those, you will need to mount the SOT-23 version on leads.
- The minimum output voltage allowed is 9V DC, while the maximum is 72V peak. This means that we should limit the AC input to 40V RMS to provide reasonable safety margins. We must ensure that the rectified output's minimum voltage does not drop below 9V during operation.

How it works

Its operation is similar to a diode bridge but with a controller IC that turns the Mosfets on when required to minimise losses. Fig.1 is the circuit diagram while Fig.2 shows how current flows during the two main phases when the bridge is conducting.

The Mosfets are arranged so the current flows from their source to drain terminals in regular operation, the opposite to a standard common-source Mosfet switch application. This is so that the current flows through the Mosfet body diodes in the forward direction.

Therefore, in the absence of the controller, current would flow through those body diodes. However, there would be a high typical 1V forward drop at high currents, similar to a silicon power diode.

Silicon Chip Ideal Rectifier Kits

SC6850 (~£18)	28mm spade version
SC6851 (~£18)	21mm square PCB pin version
SC6852 (~£18)	5mm pitch SIL version
SC6853 (~£15)	mini SOT-23 version
SC6854 (~£22)	standalone D2PAK SMD version
SC6855 (~£28)	standalone TO-220 through-hole version

During operation, the LT4320 determines which of the input voltages (IN1 & IN2) is lower and switches on either Q3 or Q4 full to connect the input terminal with the lower voltage to the negative rail and hence the negative output.

The controller switches Mosfet Q1 or Q2 on when current flows through them, reducing the effective forward voltage to about 20mV. The drop is set by the controller; if the LT4320 detects a differential greater than 20mV between the highest AC input voltage and the output terminal, it switches the respective Mosfet on harder.

If the Mosfets have a relatively high $R_{DS(ON)}$ figure resulting in more than 20mV across the Mosfet, it will be switched on fully, and the input/output differential will be higher than 20mV.

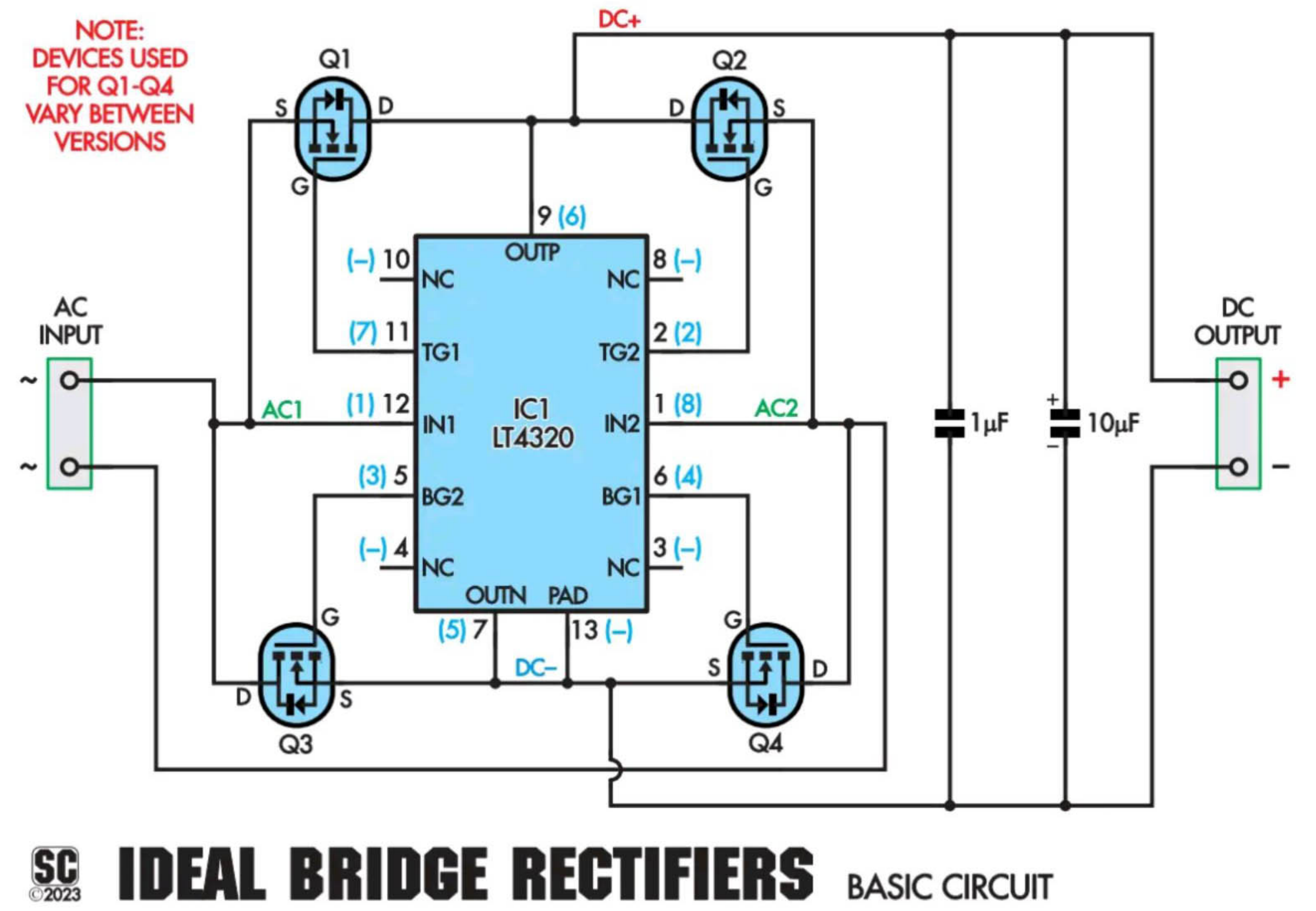

NOTE:
DEVICES USED
FOR Q1-Q4
VARY BETWEEN
VERSIONS
DC+
Q1
S
D
G
Q2
D
S
G
9 (6)
OUTP
(–) 10
NC
8 (–)
NC
(7) 11
TG1
2 (2)
TG2
AC
INPUT
~
~
AC1
(1) 12
IN1
IC1
LT4320
1 (8)
IN2
AC2
1µF
10µF
DC
OUTPUT
+
–
(3) 5
BG2
6 (4)
BG1
(–) 4
NC
3 (–)
NC
OUTN
PAD
(5) 7
13 (–)
DC–
G
D
S
Q3
G
S
D
Q4
SC
©2023
IDEAL BRIDGE RECTIFIERS
BASIC CIRCUIT

Fig.1: the circuit is slightly more complex than a conventional bridge rectifier. Pin numbers in black are for the MSOP-12 package while those in brackets in cyan are for DIP-8. Dashes in parentheses indicate pins that don't exist on the DIP-8 package.

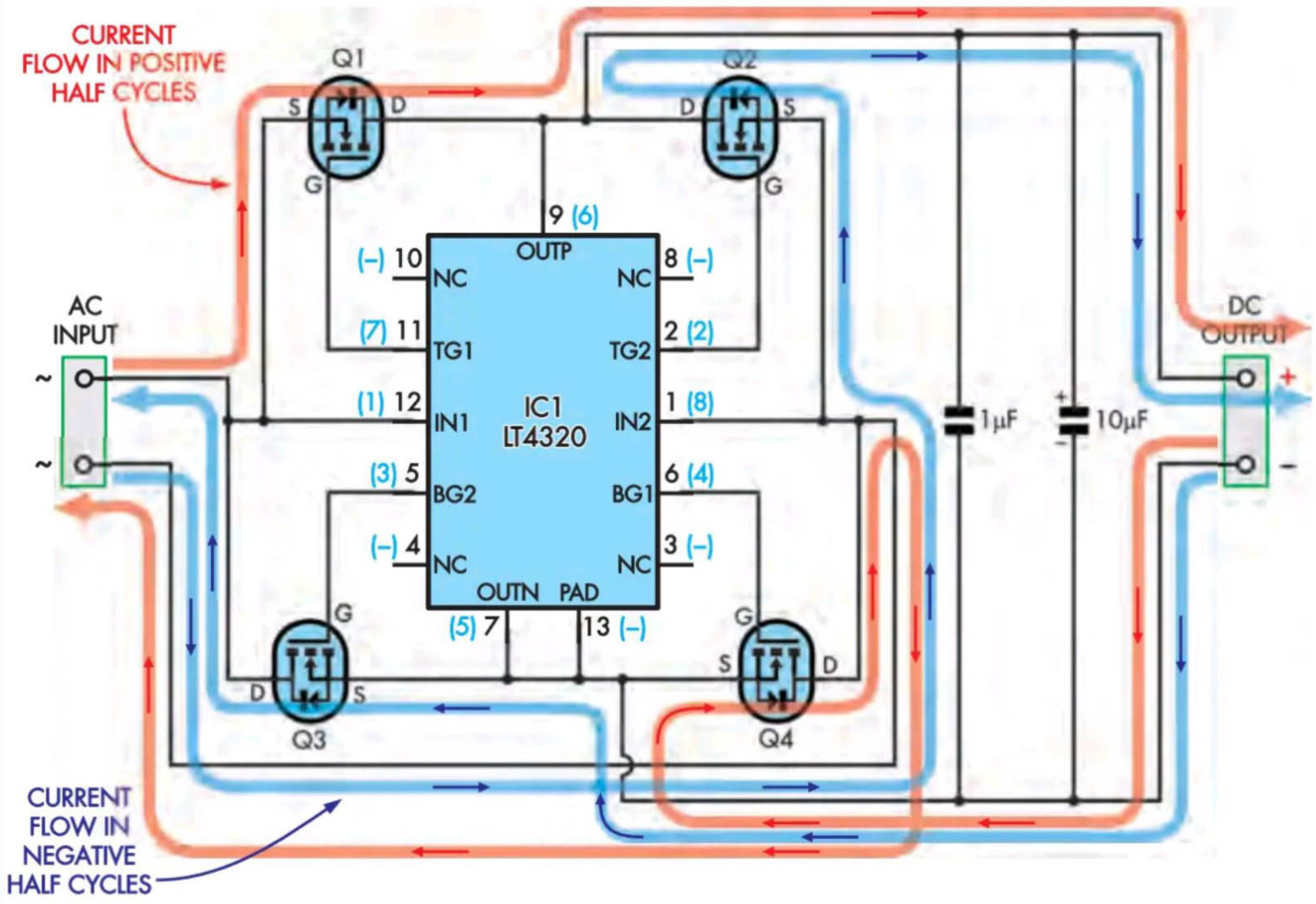

CURRENT
FLOW IN POSITIVE
HALF CYCLES
Q1
S
D
G
Q2
D
S
G
9 (6)
OUTP
(–) 10
NC
8 (–)
NC
(7) 11
TG1
2 (2)
TG2
AC
INPUT
~
~
(1) 12
IN1
IC1
LT4320
1 (8)
IN2
1µF
10µF
DC
OUTPUT
+
–
(3) 5
BG2
6 (4)
BG1
(–) 4
NC
3 (–)
NC
OUTN
PAD
(5) 7
13 (–)
G
D
S
Q3
G
S
D
Q4
CURRENT
FLOW IN
NEGATIVE
HALF CYCLES

Fig.2: during part of the mains waveform, when the upper AC input voltage is higher than the lower, IC1 switches on Q1 & Q4 and current flows via the red paths. During the opposite part of the waveform, the upper AC input voltage is lower, Q2 & Q3 are on and current flows via the blue paths.

The gate drive to the Mosfets is not very 'strong' in that a fairly low current is supplied. This reflects the application for this IC in low-frequency (50/60Hz mains) or for the LT4320-1 (to 600Hz) operation.

With a 9V DC output voltage and the top Mosfet (Q1 or Q2) V_{GS} at 2V, the pullup current is only 500µA. Our recommended DPAK SMD Mosfet, the TK6R9P08QM, has an input capacitance of 2.7nF. So the gate voltage will change at a rate of 180mV/µs.

That is terribly slow compared to most Mosfet applications, but for mains-frequency operations, if each Mosfet is on for 10% of the cycle, that's 2ms. The switch-on time of 20µs or so is only 1% of that period. The losses are minimal because this switching is just as the mains cycle crosses over.

The 1µF ceramic capacitor across the OUTP and OUTN pins is important for the correct circuit operation as it prevents the output voltage from changing too rapidly. It should be kept as close to the LT4320 as possible.

The Ideal Bridge Rectifier can operate from 9-72V. If the rectified output goes below 9V, the LT4320 will not drive the Mosfet gates, and rectification falls back to the body diodes in the Mosfets. This is OK at startup, but we must ensure the rectified rail remains above 9V afterwards. We will come back to this later on.

During tests where we were hammering the bridge and applied a load so severe that the output voltage dropped below 9V, we found that the Mosfets were getting hotter than we expected. However, that's a fairly unusual situation for a real bridge rectifier.

Parts selection

There were a few things to keep in mind when choosing the Mosfets for this design. We have tested the devices specified in the parts list and in the panel titled "Ideal Bridge Recitfier PCBs", although there is no doubt that many others would work. Besides being in the correct package for the board, they need sufficiently high voltage and current ratings, low on-resistances (for highest efficiency) and a gate-source threshold voltage in the correct range.

For the latter, the recommendation

is that it should be more than 2V. This is required to ensure that the controller can switch the Mosfet off quickly, to keep efficiency high.

Many modern Mosfets have a low gate threshold to allow them to be controlled by lower voltage circuits (often called 'logic-level' Mosfets), making them unsuitable. These can sometimes be spotted as they tend to have a lower maximum V_{GS} rating, below the ±20V to ±30V that used to be typical. However, there are still logic-level Mosfets with a higher V_{GS} rating, so you need to check the data sheet.

As for the current rating, in a bridge rectifier, the current usually only flows while the reservoir capacitors are charging. With very large capacitor banks and a low internal impedance transformer, this can be pretty short, resulting in peak charging currents much greater than the average (or "DC") current being drawn from the power supply.

The recommendation is that the Mosfets have a DC rating triple the average direct current. Therefore, we have selected Mosfets with higher current ratings than you might expect are necessary.

However, we tried not to go overboard with this as ultra-high-current Mosfets tend to have a high gate capacitance. The LT4320 does not have a strong gate drive capability, so that would slow switch-on and switch-off, resulting in increased losses.

The $V_{DS(MAX)}$ rating should be well above the voltage at which you want to operate the bridge, with a solid margin to allow for ringing and spikes. We looked for a minimum rating of 80V, although our SOT-23 version is limited to 40V.

Mosfet heating is primarily determined by the average current and their $R_{DS(ON)}$. For the TK6R9P08QM DPAK Mosfet we use in many module versions, the typical $R_{DS(ON)}$ is specified as 5.5mΩ for $V_{GS} > 10V$.

The LT4320 delivers about 11V to the gates for voltages greater than 13V. For an average current of 10A, this results in 550mW dissipation in each conducting Mosfet, or 275mW per Mosfet for an AC input, which is easily manageable, and the boards only get warm.

For a current of 20A, this dissipation increases to about 1W per Mosfet, making them very warm indeed, at

Ideal Bridge Rectifier PCBs

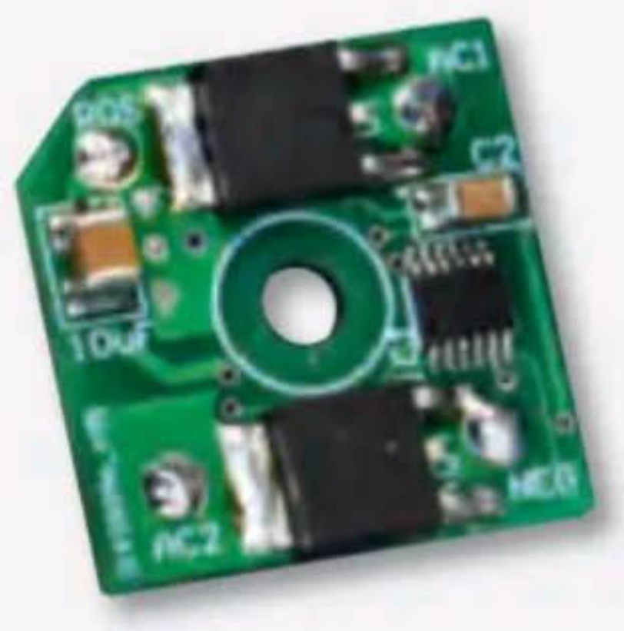
For maximum flexibility, we have produced six different PCBs that implement essentially the same circuit, as follows:

#1 Square 28mm metal bridge using 6.3mm spade connectors

- *Compatible with KBPC3504*
- PCB code: 18101241 (28 × 28mm with a central mounting hole)
- Current & voltage handling: 10A continuous (20A peak), 72V
- Connectors: 6.3mm spade lugs, 18mm tall
- IC1 package: MSOP-12 (SMD)
- Mosfets: TK6R9P08QM,RQ (DPAK/TO-252 SMD)

Operates at 10A continuously and much higher currents intermittently but will get hot. In a long-term 8A test, it reached 79°C in free air.

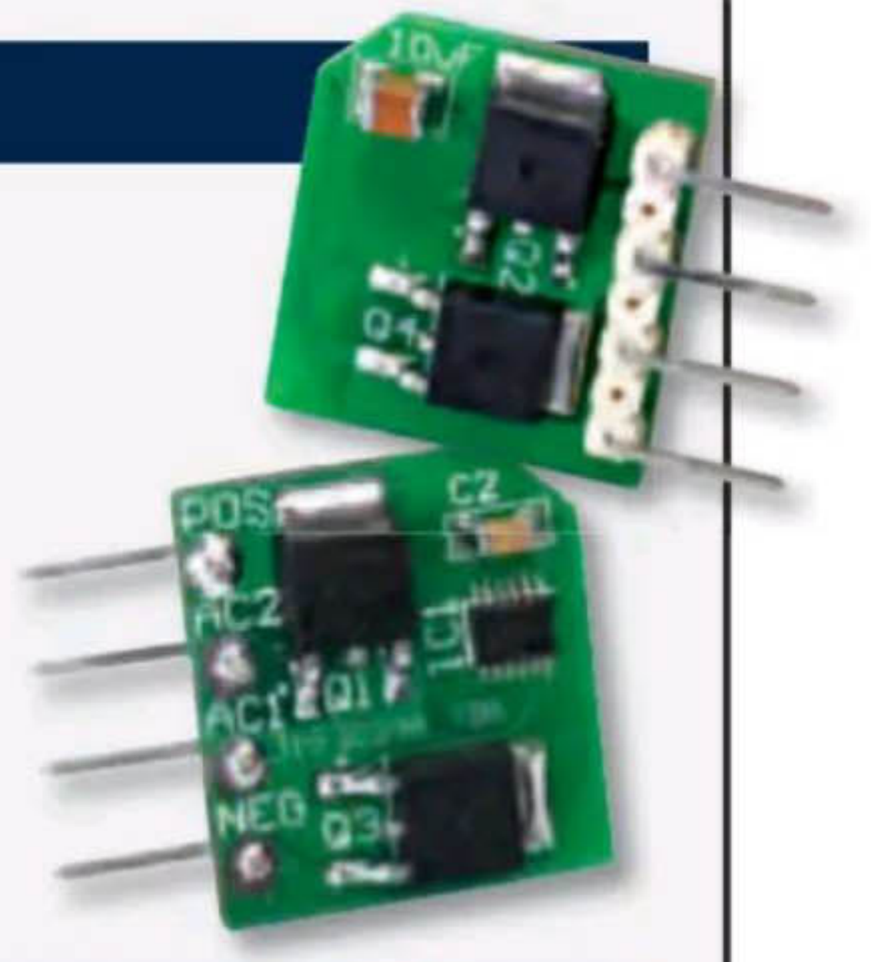
#2 Square 21mm plastic bridge with 13mm pitch pins



- *Compatible with PB1004*
- PCB code: 18101242 (22 × 22mm with a central mounting hole)
- Current & voltage handling: 10A continuous (20A peak), 72V
- Connectors: solder pins on a 14mm grid (can be bent to a 13mm grid)
- IC1 package: MSOP-12 (SMD)
- Mosfets: TK6R9P08QM,RQ (DPAK/TO-252 SMD)

A PB1004 leaded bridge replacement, typically capable of 5-10A. We used these to upgrade our Dual Hybrid Power Supply module.

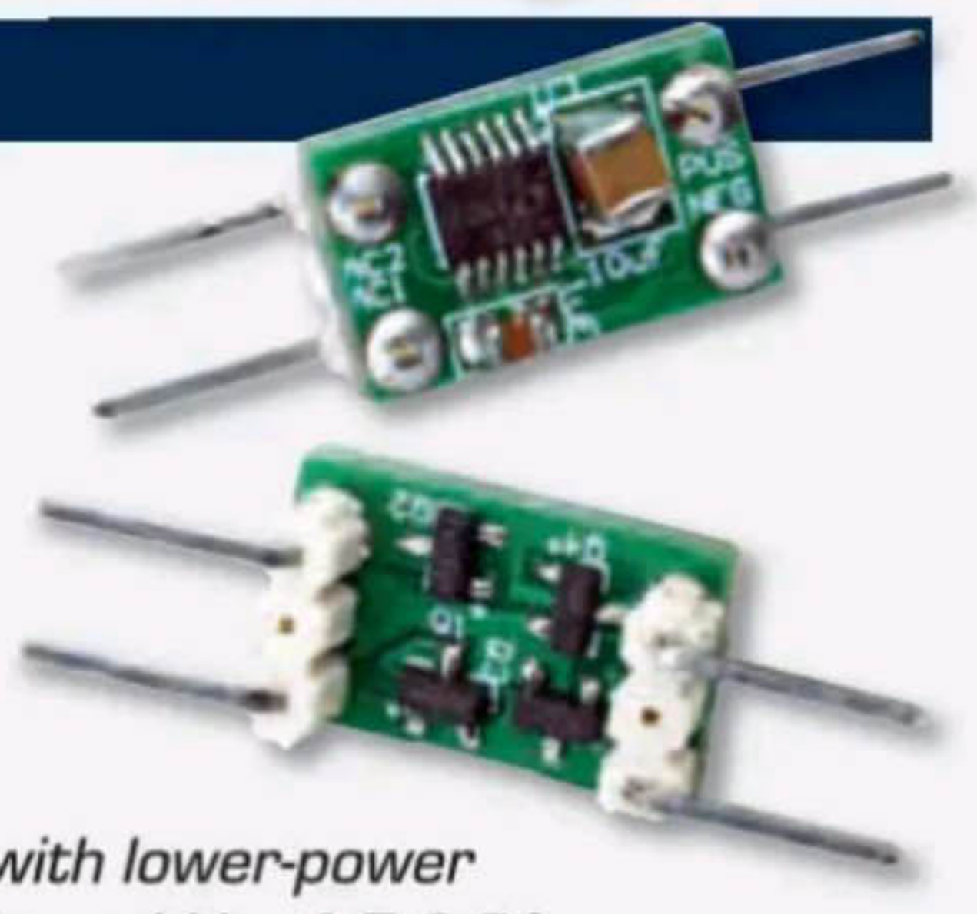
#3 5mm pitch SIL



- *Compatible with KBL604*
- PCB code: 18101243 (23 × 20mm)
- Current & voltage handling: 10A continuous (20A peak), 72V
- Connectors: solder pins at 5mm pitch
- IC1 package: MSOP-12 (SMD)
- Mosfets: TK6R9P08QM,RQ (DPAK/TO-252 SMD)

The 5mm pitch SIL bridge rectifier drop-in replacement module.

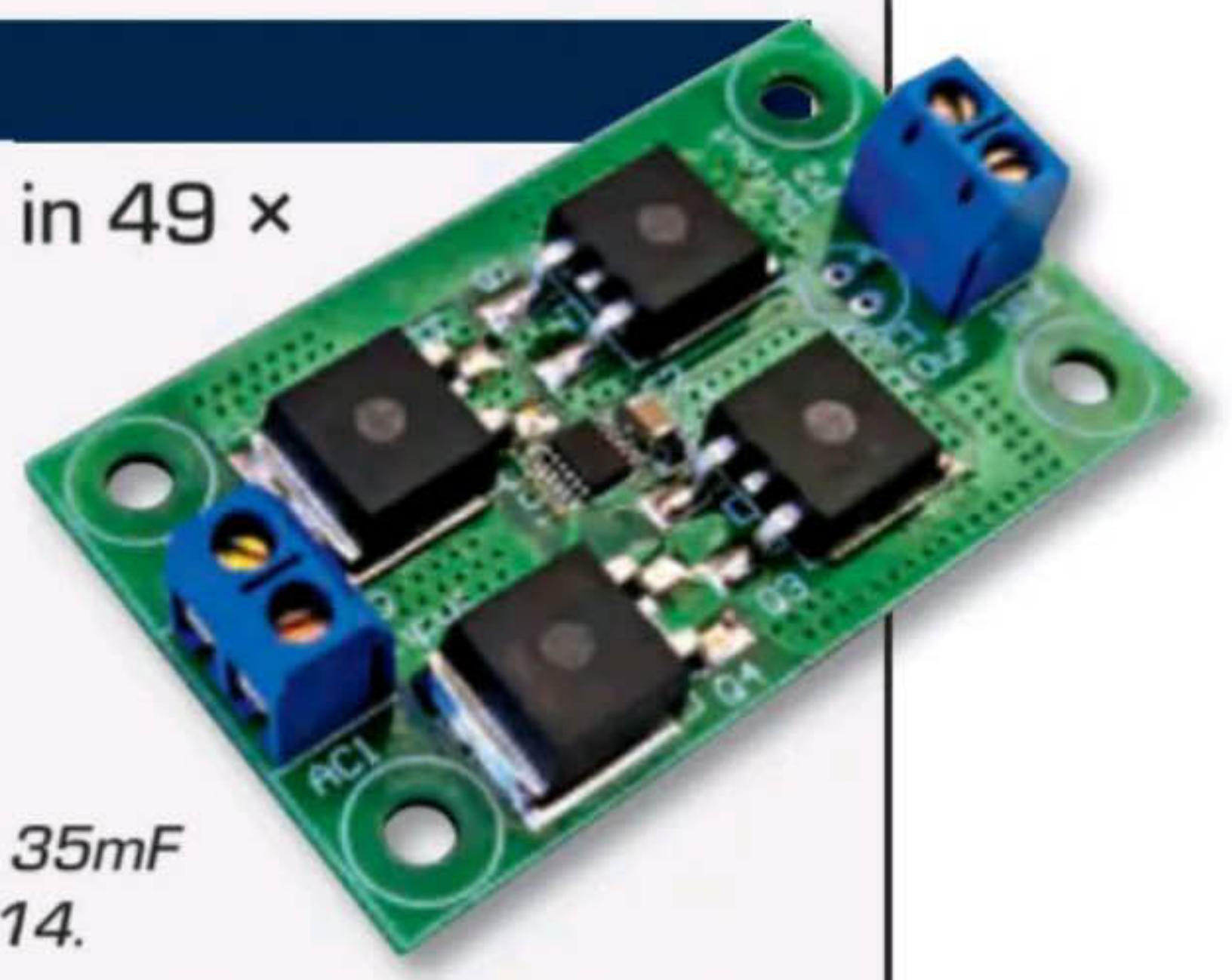
#4 Tiny inline bridge



- *Width of W02/W04*
- PCB code: 18101244 (9 × 15mm)
- Current & voltage handling: 2A continuous, 40V
- Connectors: solder pins 5mm apart at either end
- IC1 package: MSOP-12 (SMD)
- Mosfets: SI2318DS-GE3 (SOT-23 SMD)

The baby of the crew, the SOT-23 based version optimised for putting inline with lower-power circuits. These Mosfets are rated at 40V & 3.9A, but we reckon a safer limit would be 1.5-2.0A.

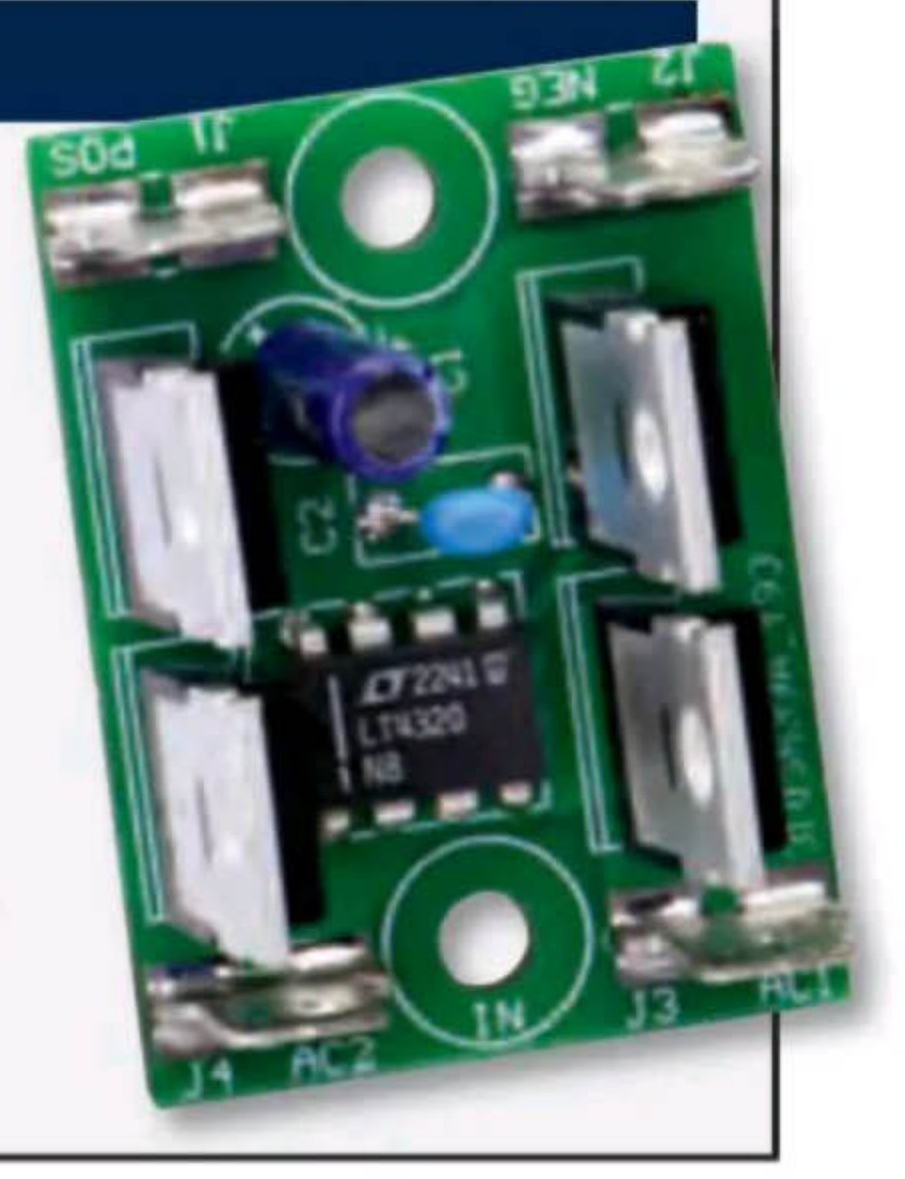
#5 Standalone SMD version



- PCB code: 18101245 (59 × 36mm with mounting holes in 49 × 26mm rectangle)
- Current & voltage handling: 20A continuous, 72V
- Connectors: 5mm screw terminals at either end
- IC1 package: MSOP-12 (SMD)
- Mosfets: IPB057N06NATMA1 (D2PAK/TO-263 SMD)

The D2PAK version, which I have tested for half an hour at 12V AC and 8A (into a 35mF capacitor with a 2Ω load across it). You can see this being stress tested on page 14.

#6 Standalone through-hole version



- PCB code: 18101246 (38 × 28mm with 70µm-thick copper and mounting holes 29mm apart)
- Current & voltage handling: 40A continuous, 72V
- Connectors: 6.3mm spade lugs, 18mm tall
- IC1 package: DIP-8 (through-hole)
- Mosfets: TK5R3E08QM,S1X (TO-220 through-hole)

The TO-220 version is a bit of a beast and, along with the D2PAK version shown above, it will easily handle 8-10A RMS continuously. It uses a DIP-8 controller IC and allows you to mount a heatsink to the Mosfets if you want to rectify some serious currents. All the images here are not shown to scale.

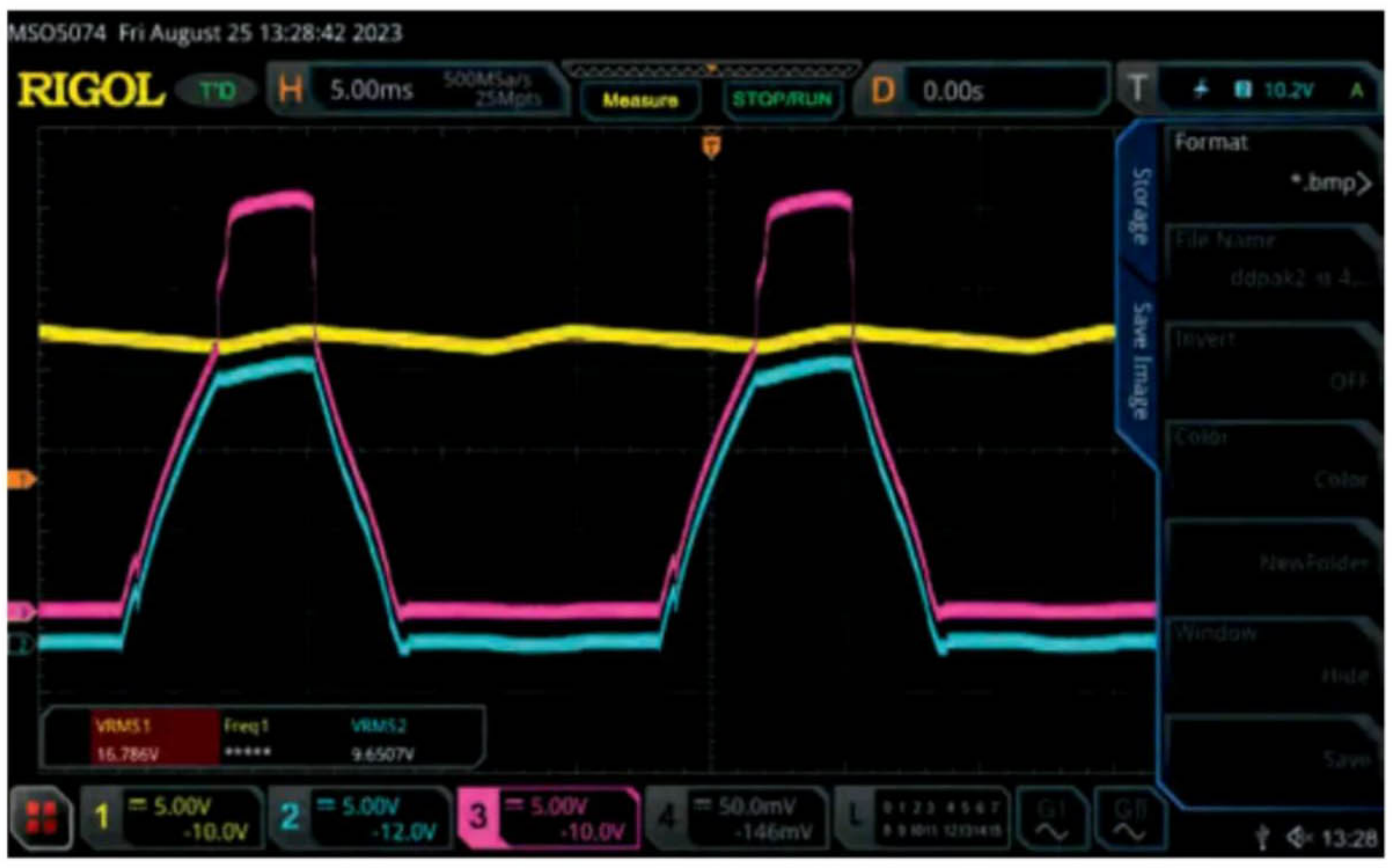

MSO5074 Fri August 25 13:28:42 2023
RIGOL
H 5.00ms
D 0.00s
Format
Storage
Save Image
VRMS1 16.786V
VRMS2 9.6507V
1 5.00V -10.0V
2 5.00V -12.0V
3 5.00V -10.0V
4 50.0mV -146mV
13:28

Fig.3: the cyan trace is the positive portion of the incoming AC waveform, yellow is the filtered DC output, while mauve is the positive Mosfet gate drive. The cyan AC trace is offset by -2V; otherwise, the mauve trace would obscure it much of the time.

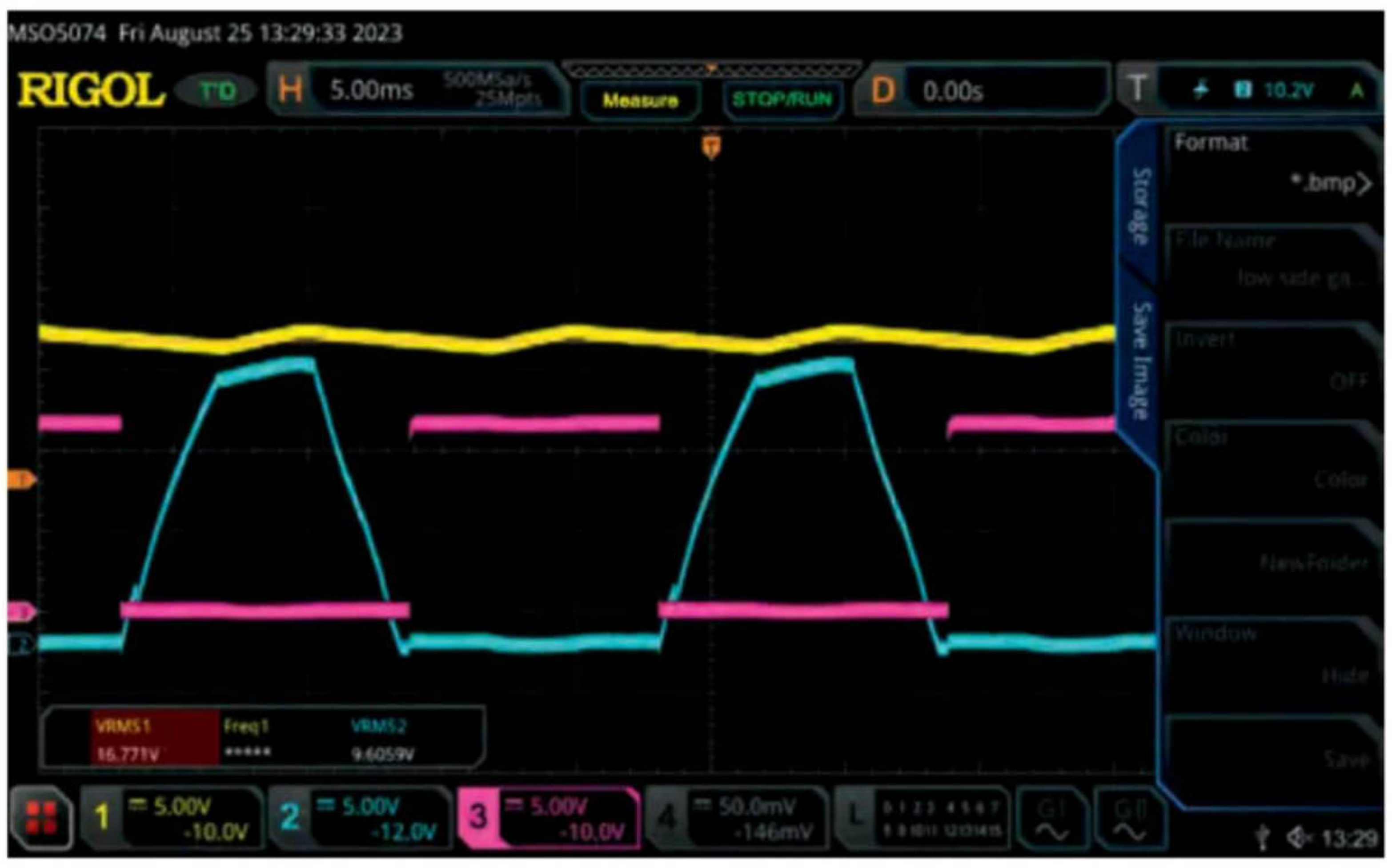

MSO5074 Fri August 25 13:29:33 2023
RIGOL
H 5.00ms
D 0.00s
Format
Storage
Save Image
VRMS1 16.771V
VRMS2 9.6059V
1 5.00V -10.0V
2 5.00V -12.0V
3 5.00V -10.0V
4 50.0mV -146mV
13:29

Fig.4: a similar setup to Fig.2, but this time, we're monitoring the gate of one of the low-side Mosfets (mauve). You can see how it's switched on with a duty cycle close to 50%, synchronised with the zero crossings of the AC waveform.

which point you should consider building the TO-220 version.

The recommended TO-220 Mosfet has an $R_{DS(ON)}$ of 4.2mΩ at full drive and, at 40A, will drop 160mV; it would be closer to 1.2V in a regular bridge at this sort of current. The power dissipation in each Mosfet would be 3.5W for an AC input, which is significant but manageable with small heatsinks. In this case, a regular diode-based bridge would get toasty, as it would dissipate 48W per diode!

The LT4320 IC comes in an SMD (MSOP-12) and through-hole (DIP-8) version. These are available from all the major component suppliers and is included in the Silicon Chip kits.

For the Mosfets, we have tried to stick to standard parts, with DPAK (TO-252) being our overall preference as they are large enough to handle a decent amount of dissipation (~1W) without being so large that they take up a lot of space. The other Mosfets we've used come in TO-220 packages (for really high current applications) and the tiny SOT-23 (for when space is tight).

By sticking to these standard footprints, you can use alternative parts if necessary.

PCB design

Most of the modules we present use surface-mounting parts to fit into the space we have. We have also resorted to placing components on both sides of the PCB, as doing that was essential to match some of the common bridge rectifier form factors.

For higher-current modules, we need to be conscious of the current rating of the PCB traces. To fit the parts onto the KBPC3504 form-factor board, along with the very wide tracks that a 30-40A rating warrants, is quite a challenge. Our version manages to keep all high-current tracks short and thick, but that forced the layout to be slightly larger than the original rectifier.

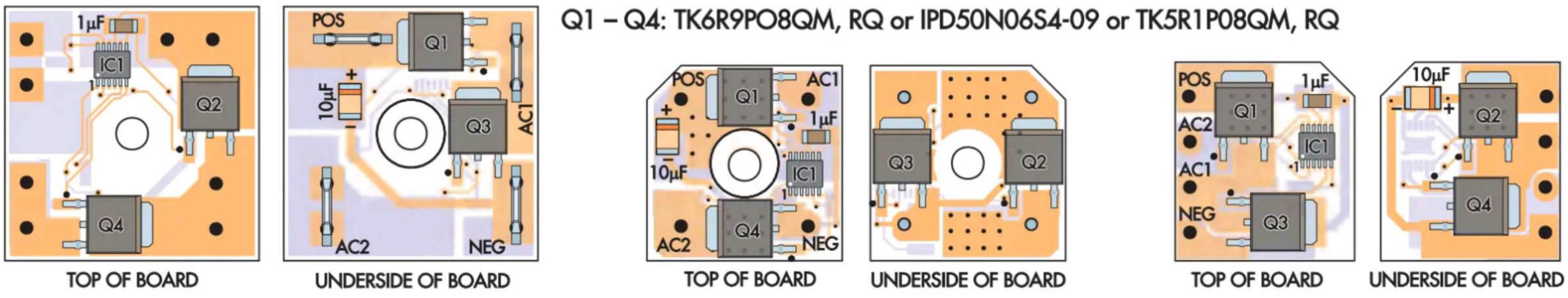
There is no specific 'rating' for PCB traces; there are guidelines, but too many variables exist to realistically put a simple, accurate number to a track width. Still, the voltage drop and resulting heating of the track must be considered. In the limiting case, tracks can fuse or melt.

We have specified '2oz' (70μm thick) copper traces on the TO-220 PCB, twice as thick as a standard '1oz' (35μm) PCB. This will halve resistive losses in the PCB at the price of it being a lot harder to solder due to the thick copper acting as a heatsink (although that will have benefits during operation, drawing heat away from components faster).

It is evident that at high currents, even an 'ideal diode' warrants careful attention to power ratings, losses and dissipation. But these are reduced to a level where a practical solution can be developed. We recommend that you pay careful attention to losses and heat if you use this at really high currents. At least verify that the chosen module doesn't get overly hot at your expected maximum current draw.

Waveforms & verification

Figs.3 & 4 show the input, output and gate drive waveforms for the Ideal Bridge Rectifier operating at 4A RMS. Note that the AC input is offset -2V


Q1 – Q4: TK6R9PO8QM, RQ or IPD50N06S4-09 or TK5R1P08QM, RQ
1μF
IC1
Q2
Q4
TOP OF BOARD
POS
Q1
10μF
Q3
AC1
AC2
NEG
UNDERSIDE OF BOARD
POS
AC1
Q1
1μF
10μF
IC1
AC2
Q4
NEG
TOP OF BOARD
Q3
Q2
UNDERSIDE OF BOARD
POS
1μF
Q1
AC2
IC1
AC1
NEG
Q3
TOP OF BOARD
10μF
Q2
Q4
UNDERSIDE OF BOARD

Figs.6-11: use these overlay diagrams to guide the component placement on each version. The four smaller PCBs have components on both sides. Generally, it's best to fit all the SMDs on one side, then all the SMDs on the other, then any remaining through-hole parts. Note that while we've specified non-polarised ceramic 10μF capacitors for the first four variants, tantalums are shown in case you want to use them, in which case they must be orientated as shown.

to allow a clearer view – there is so little voltage drop across the Mosfet that the output visually 'tracks' the input AC much of the time. The gate drive is over 10V, so the Mosfet is switched fully on.

To illustrate the low voltage drop across the power Mosfet even at 4A, Fig.5 shows the input and output waveforms with no offset.

Having built the modules, we decided to run some extreme tests as we didn't want our readers to make them only to have them blow up! We loaded the 28mm bridge design (KBPC3504 compatible) to draw 5A RMS from a toroidal transformer and left it running for several hours. The Ideal Rectifier stabilised at 42°C. Ramping the current to 8A led to it reaching 72°C, which is not unreasonable for the current.

Swapping in a regular KBPC3504 at 4A without heatsinking, it reached 79°C after a few minutes.

As shown earlier, we 'upgraded' our Dual Hybrid Power Supply with Ideal Rectifiers, which saves 10W of heat per board at full output or 20W in total. For this, we used the PB1004 format modules and soldered them on leads directly to the PCB, as at 5A, they do not get hot enough to demand a heatsink.

During testing, we had a test setup with a 12V AC output transformer, an Ideal Bridge Rectifier and a 22mF capacitor. Things were going great until we reduced the load resistance to somewhere near 1Ω, and the output voltage dropped below 9V due to the capacitor discharging between cycles.

The LT4320 stopped driving the Mosfets, and instead of there being 20mV across them, there was suddenly about 1V across the body diodes at about 15A. The smoke quickly escaped from the DPAK Mosfets. We recommend that you avoid that situation.

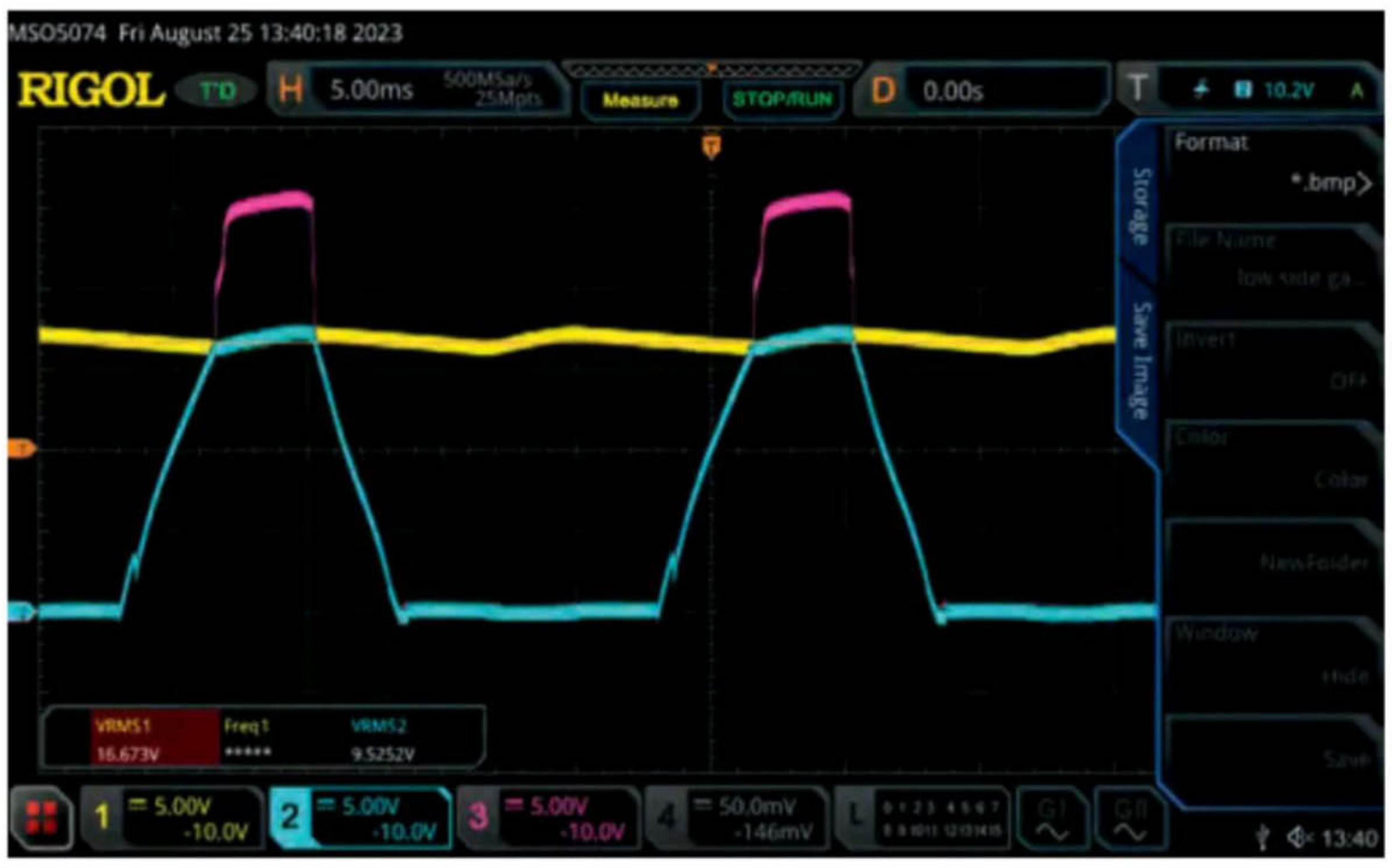

MSO5074 Fri August 25 13:40:18 2023
RIGOL
T'D
H 5.00ms
500MSa/s 25Mpts
Measure
STOP/RUN
D 0.00s
T
10.2V
A
Storage
Save Image
Format
*.bmp
File Name
low side ga...
Invert
OFF
Color
Color
NewFolder
Window
Hide
Save
VRMS1
16.673V
Freq1
*****
VRMS2
9.5252V
1 5.00V -10.0V
2 5.00V -10.0V
3 5.00V -10.0V
4 50.0mV -146mV
13:40

Fig.5: the same waveforms as in Fig.2 but without the -2V offset on the AC input. The IPP083N10N Mosfets on this board stabilised at 38°C in the lab. The dummy load, on the other hand, measured 130°C.

Construction

With so few parts on the board, construction is straightforward. Refer to the PCB overlay diagram(s) for whichever version(s) you are building, shown in Figs.6-11.

The principal challenge is that for all but the TO-220 version, we're using the LT4320 IC in an MSOP-12 package with a thermal pad on its base. This thermal pad makes this part a tad harder to solder than your average SOIC/SOP SMD part. There are two (or three) practical soldering options:

1. Using a reflow oven. If you already own one of these, chances are you are all over how to mount the part. Each oven has its own quirks, so we will leave this to you.

2. Use a toaster oven as a 'bodge'. You can read articles on turning a toaster oven into a reflow oven (April & May 2021), but there is also a 'quick and dirty' method that works.

Buy a super-cheap toaster oven (we often see these for sale under £25) and stick a K-type thermocouple alongside your board. Apply solder paste to the pads and carefully place the parts on top. Preheat the PCB to 100°C in the oven, then turn the oven up to maximum. Watch closely until the temperature hits 220°C. At this point, you should have seen the solder flow. Immediately turn the oven off and open the door.

3. Use a hot air gun. That is how we built all the prototypes, to convince ourselves that it would work for you (see the photo overleaf). Even though we have a reflow oven, we often use the hot air gun as it is quick and easy (they're also surprisingly inexpensive). We used this technique just for the LT4320, leaving the easier capacitors and Mosfets to be hand-soldered. The key steps are:

a | Apply a small amount of solder paste to each pad and the central thermal pad. Do not overdo this; a modest smear is sufficient. We use 60/40 tin/lead solder paste as it melts at a lower temperature, making it generally easier to work with. Nothing is stopping you from using lead-free solder, but remember that it requires higher temperatures.

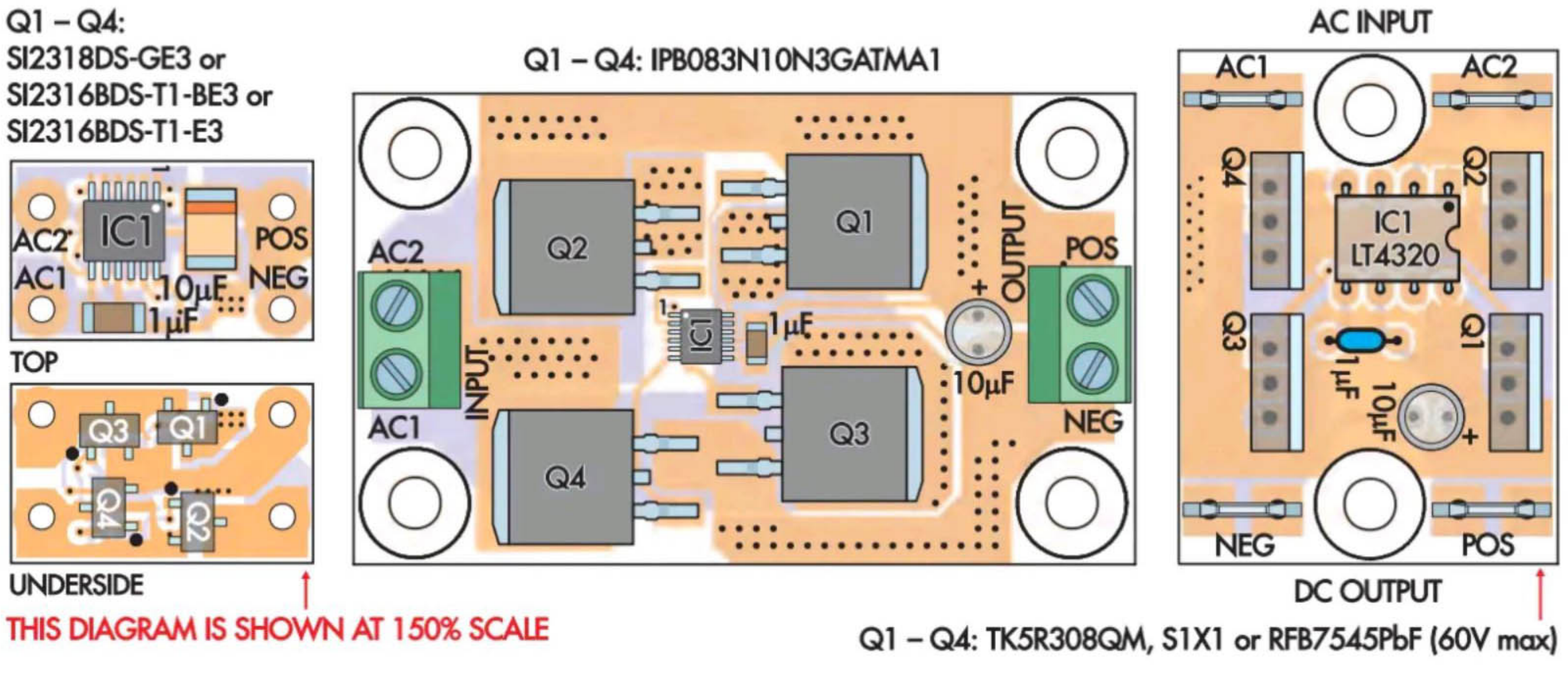
b | Place the LT4320 using tweezers. There should be sufficient solder paste to stick in place, but not so much that it squishes everywhere.

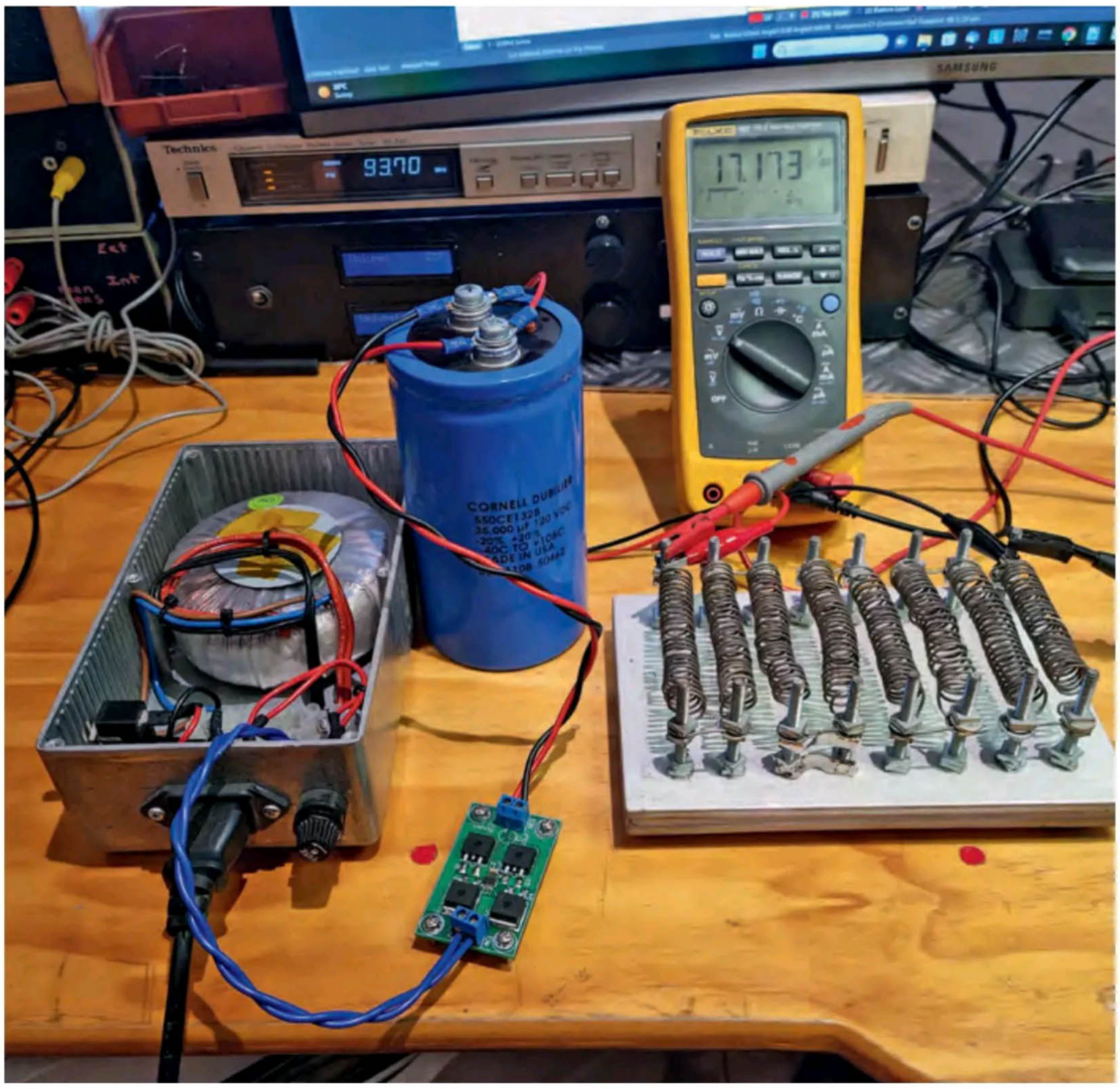
c | Check that the LT4320 is the right way around. Double-check, as this is by far the most expensive part in this project.

d | Put the board on a heat-resistant surface, such as a PCB off-cut. Do not use your desk as it will get quite hot!

e | Set your hot air gun to about 300°C.

f | Apply heat to the board in a gentle waving motion from about 15cm away, so the board around the IC is heated reasonably evenly. We want to preheat the board to something in the region of 100°C over a minute or so.


Q1 – Q4:
SI2318DS-GE3 or
SI2316BDS-T1-BE3 or
SI2316BDS-T1-E3
IC1
AC2
AC1
POS
NEG
10μF
1μF
TOP
Q3
Q1
Q4
Q2
UNDERSIDE
THIS DIAGRAM IS SHOWN AT 150% SCALE
Q1 – Q4: IPB083N10N3GATMA1
AC2
AC1
INPUT
Q2
Q4
Q1
Q3
IC1
1μF
10μF
+ OUTPUT
POS
NEG
AC INPUT
AC1
AC2
Q4
Q2
IC1
LT4320
Q3
Q1
1μF
10μF
NEG
POS
DC OUTPUT
Q1 – Q4: TK5R308QM, S1X1 or RFB7545PbF (60V max)



My poor wirewound nichrome dummy load reached 320°C while the Mosfets on the D2PAK standalone module only reached 67°C.

Soldering the MSOP-12 LT4320 IC using a low-cost hot air 'rework' station. These are invaluable for all sorts of jobs; they make it especially easy to desolder SMDs. In this case, the killer feature is the ability to heat the IC enough to solder the pad underneath.

g | Once the board is well warmed up, bring the hot air gun to about 5-10cm from the board and work around the IC. Have your tweezers handy; if the IC moves a lot, you might need to nudge it back into position. Having said that, surface tension will typically pull it into place if you're blowing the air directly from above.

h | Watch the solder paste. As the board approaches 220°C, you will see the paste changing from dull granular material to a shiny liquid. The change is significant, so you shouldn't miss it.

i | As the solder melts, it also creates a lot of surface tension and will pull the IC into position.

j | Do not overheat the board. Once all the solder has reflowed, take the heat gun away.

k | Allow the board to cool naturally. Do not put any liquid on the board to accelerate the cooling.

l | You might see several pins with solder bridges across them. Fold some solder wick across the tip of your iron and 'dab' the pins to melt the bridge into the wick. Adding a little flux paste to the bridge first usually helps. With a little practise, this is quick and easy. We get quite a few bridges to fix as we are too generous with the solder paste!

For the remaining SMD parts, a regular soldering iron works fine. We generally tack down one of the SMD leads and make sure the part is straight. For the two-pin passives, all that's left is to solder the second lead.

For the Mosfets, apply the iron to the source (main tab) at the junction of the tab and the PCB pad. Put a small amount of solder between the iron and the tab and wait until the solder flows.

Once both the pad and the component lead are hot, the solder will flow freely under the component. After that, you can solder the remaining small pins.

The 6.3mm spades, screw connectors or wire leads are through-hole parts, so solder them as usual.

Testing

Testing the Ideal Bridge Rectifier is not complex and can be undertaken at low power.

First, connect a 220Ω 1W resistor across the output, or an alternative resistor with a power rating that can withstand the DC voltage we will apply in the following steps. Connect a multimeter across the test resistor with the meter's positive line to the positive output of the ideal bridge rectifier.

Connect a 12V DC power supply to the input of the Ideal Bridge Rectifier and verify that the output gives a +12V reading on the meter. Verify that the voltage drop is less than 100mV.

Then swap the polarity of the input voltage and verify that the output is still giving a +12V reading on the meter, and the voltage drop is still less than 100mV.

If it does not work

- Check all solder connections.
- Check the orientation of the LT4320 IC.
- If using TO-220 Mosfets, check their orientations.
- If building the through-hole board, check the orientation of the electrolytic capacitor.
- Check your test setup; is the power supply in current limiting? Check the input voltage.

Using it

Among the six different modules, you will likely find a 'drop in' solution. The SIL and 19mm pin bridges should solder straight to a PCB that's designed for a regular bridge rectifier.

For an audio amplifier, you would ideally mount two of the standalone versions in the chassis and run individual windings to each.

Remember that the LT4320 operates from 9V to 72V. If your output voltage falls below this, the LT4320 will not drive the Mosfets, and the bridge will only operate using the body diodes.

That is OK to get the circuit started, but at high currents, the dissipation can be very high.

This is only a concern if your design uses low rail voltages, or you are likely to do something as silly as we did and drive the rectifier so hard that your capacitor discharges massively between 50Hz cycles. That won't happen in a typical power supply or power amplifier.

Conclusion

The Ideal Diode Bridge Rectifier can significantly improve the efficiency of just about any circuit that requires a rectifier for only a modest increase in the device's overall cost.

Best of all, for devices like power supplies and audio amplifiers, you can get even more output voltage or power than you would with a standard diode-based rectifier. Less heat from the rectifier also means less cooling is required.

Don't forget, though, that for applications like an audio amplifier with split rails (positive and negative), unlike a diode-based rectifier, you will need two of these devices, one for each supply rail.

The transformer also needs to have two separate secondary windings. That's because the control chip only monitors the voltage across the upper two Mosfets.

With six different designs in a range of sizes, current and voltage ratings, you're bound to find one that suits your application. PE

Parts List – Ideal Diode Bridge Rectifier

Common parts for versions #1 to #4 (from Mouser, DigiKey or Farnell)
1 LT4320IMSE#TRPBF ideal bridge controller IC, MSOP-12 (IC1)
1 1μF 100V X7R M3216 SMD ceramic capacitor **[CL31B105KCHNNNE]**
1 10μF 100V X7S M3225 SMD ceramic capacitor **[GRM32EC72A106KE5K]**

#1 28mm spade version
1 double-sided PCB coded 18101241, 28 × 28mm
4 6.3mm PCB-mounting vertical spade connectors
4 TK6R9P08QM,RQ, IPD50N06S4-09 or TK5R1P08QM,RQ
N-channel Mosfets, DPAK/TO-252 (Q1-Q4)

#2 21mm square PCB pin version
1 double-sided PCB coded 18101242, 22 × 22mm
1 10cm length of 1.5mm diameter tinned copper wire
4 TK6R9P08QM,RQ, IPD50N06S4-09 or TK5R1P08QM,RQ
N-channel Mosfets, DPAK/TO-252 (Q1-Q4)

#3 5mm pitch SIL version
1 double-sided PCB coded 18101243, 23 × 20mm
1 10cm length of 1.5mm diameter tinned copper wire
4 TK6R9P08QM,RQ, IPD50N06S4-09 or TK5R1P08QM,RQ
N-channel Mosfets, DPAK/TO-252 (Q1-Q4)

#4 Mini SOT-23 version
1 double-sided PCB coded 18101244, 9 × 15mm
1 10cm length of 1.5mm diameter tinned copper wire
4 SI2318DS-GE3, SI2316BDS-T1-BE3 or SI2316BDS-T1-E3
N-channel Mosfets, SOT-23 (Q1-Q4)

#5 Standalone D2PAK SMD version
1 double-sided PCB coded 18101245, 59 × 36mm
2 mini horizontal terminal blocks, 5mm or 5.08mm pitch
1 LT4320IMSE#TRPBF ideal bridge controller IC, MSOP-12 (IC1)
1 1μF 100V X7R M3216 SMD ceramic capacitor **[CL31B105KCHNNNE]**
1 10μF 100V radial electrolytic capacitor, 2.5mm pitch, ≤6.3mm diameter
[Kemet ESL106M100AE3AA]
4 IPB083N10N3GATMA1 N-channel Mosfets, D2PAK/TO-263 (Q1-Q4)
[ESL106M100AE3AA]

#6 Standalone TO-220 through-hole version
1 double-sided PCB coded 18101246, 38 × 28mm, with 70μm-thick copper
4 6.3mm PCB-mounting vertical spade connectors
1 LT4320IN8#PBF ideal bridge controller IC, DIP-8 (IC1)
4 TK5R3E08QM,S1X (80V) or RFB7545PbF (60V)
N-channel Mosfets, TO-220 (Q1-Q4)
1 1μF 100V X7R radial ceramic capacitor, 5mm pitch
[RDER72A105K2M1H03A]
1 10μF 100V radial electrolytic capacitor, 2.5mm pitch, ≤6.3mm diameter
[Kemet ESL106M100AE3AA]

Practically Speaking

Hands-on techniques – by Jake Rothman

A shocking experience!

I had a bad electric shock the other day. I hadn't suffered one for years, so complacency had obviously set in.

A friend had brought in his Yamaha electric motorbike charger and battery. The bike can be used on motorways, so the battery is huge, along with its associated switch-mode power supply unit/charger.

The SMPSU was giving no output, so I took it apart. My friend assured me he hadn't plugged it in for a week.

Looking inside, I found that it was a common SMPSU problem: bad lead-free soldered joints around the enamelled wire from the ferrite transformer. This usually happens when the enamel hasn't been fully removed before soldering.

Also, lead-free solder has inferior wetting properties and its brittleness results in cracking from the high-frequency vibration due to magnetostriction in operation.

I soon set to work, scraping off the burnt flux with a scalpel around the joints to get a good look. **BANG!**

I got a massive DC belt from one arm to the other, very nasty, like an old Fender valve amp HT rail but worse. The scalpel was nowhere to be seen; luckily it wasn't embedded in my friend's head!

I shouted "that felt like bloody 350 volts". I got my meter out and shakily measured between the pin I was scraping and the chassis, and it was.

I thought (belatedly) that I had better discharge the main smoothing capacitors, of which there were three in parallel. So, stupidly, I got my nice insulated Bahco Ergo pliers out and shorted the pins. **BANG!** It blew one of the tips off.

4500µF of capacitance charged to 350V is a lot of energy ($E=CV^2$ so 550J)! Thank god I had urinated not long before I undertook this procedure!

Still shaking a bit, I continued and fixed the joints. I wasn't going to be beaten by this modern 'disposable' electronics. Having fixed the bad joints, I soldered a bleeder resistor of 39kΩ 5W across the caps and switched it on.

The LEDs lit up and it gave the correct 80V DC output at 10A; perfect. My friend thought the whole thing was most entertaining! It's a good job it worked. He's now enjoying his bike and I found the scalpel stuck in the skirting board a week later.

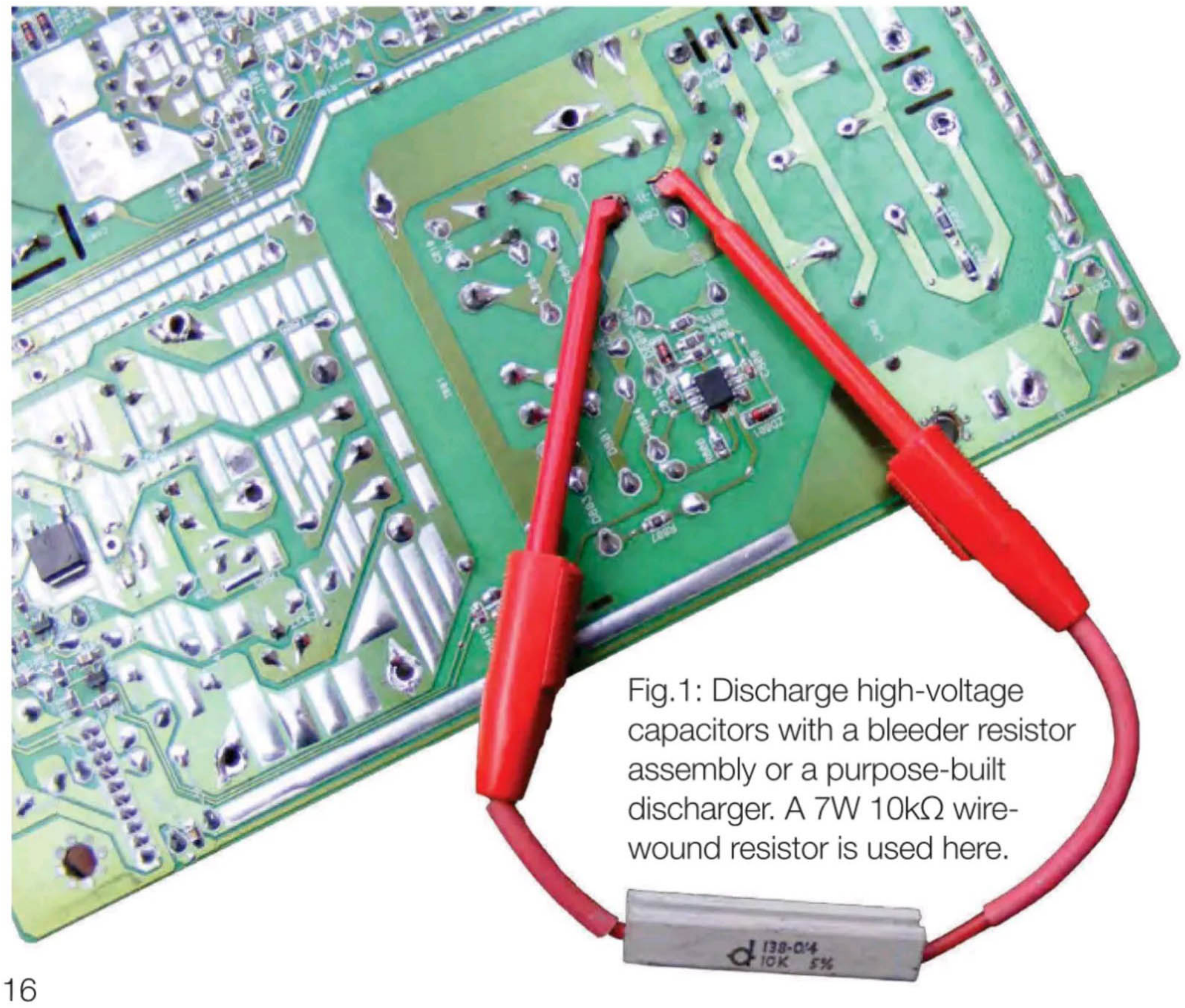


Fig.1: Discharge high-voltage capacitors with a bleeder resistor assembly or a purpose-built discharger. A 7W 10kΩ wire-wound resistor is used here.

Morals of the story

- Don't assume something is discharged, even if the client says it hasn't been switched on for a week!
- Don't use a metal Swann Morton scalpel for repair work; use a plastic-handled one instead.
- Don't assume all SMPSUs have bleeder resistors wired across the main smoothing capacitor bank. Even if it has, the resistor may have become open circuit. This circuit didn't have one because the continuous dissipation would be high and reduce its efficiency. *Editor's note: clever designers can add bleeders that are out of circuit when mains power is applied.*
- Measure the voltage across big capacitors as soon as you open the case and before you start working on the PCB. If they are charged, discharge them slowly with a bleeder resistor attached to insulated test probes, as shown in Fig.1. I use the retractable probes with an internal spring with extra rubber sleeving on all accessible parts (Fig.2). The retractable probes can clip directly onto the capacitor leads protruding from the underside of the PCB, as shown in Fig.3. Alternatively, the capacitor can be discharged above the board as illustrated in Fig.4.
- Remember the 'left hand in pocket' rule; don't make an easy current route through your heart. If you must hold the metal enclosure or chassis while working on it, physically clamp it or insulate your hand. Even relatively mild shocks involving the heart can lead to cardiac arrhythmias.
- Have a pee before working on high-voltage circuits. There's nothing more embarrassing than getting a small shock and wetting yourself, then getting a massive one because you are standing in a pool of urine. I know because I once did it in front of a load of students! It became a standing joke; I defused it by having a spare pair of M&S underpants in the first aid box. I said if anybody needed them, they would be £12.00 plus a clean-up fee.

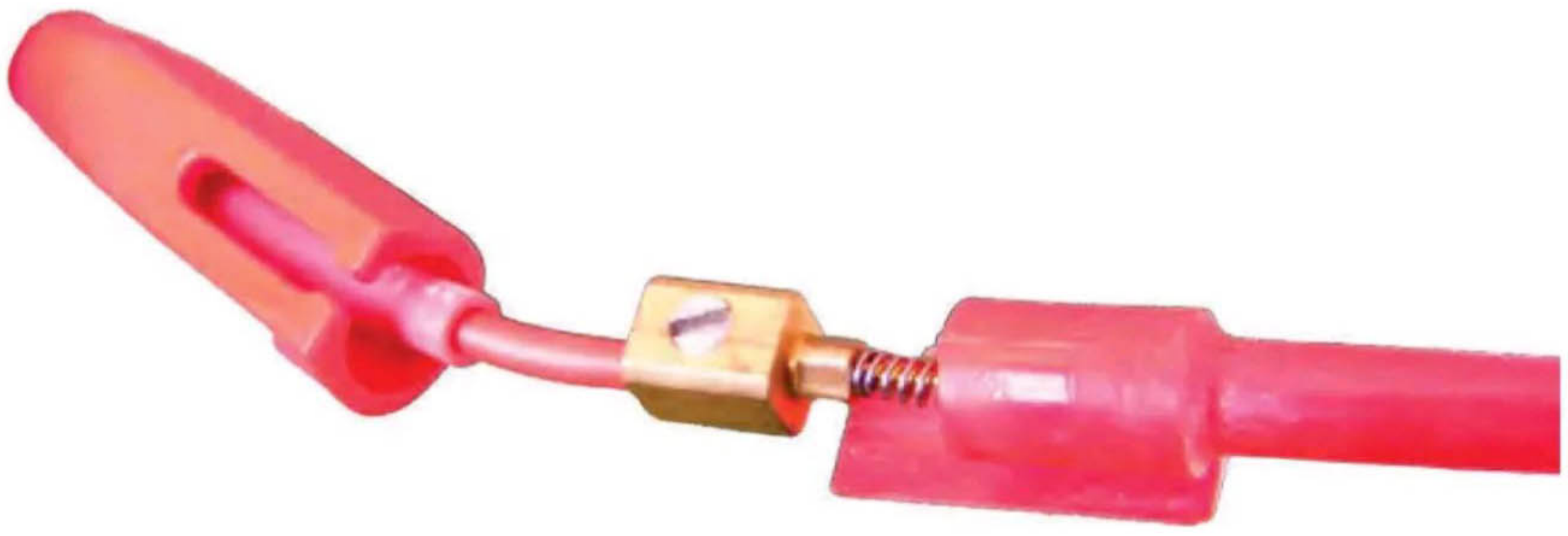


Fig.2: Some retractable test probe assemblies use a small screw to attach the wire. These are some old gripping ones from RS.

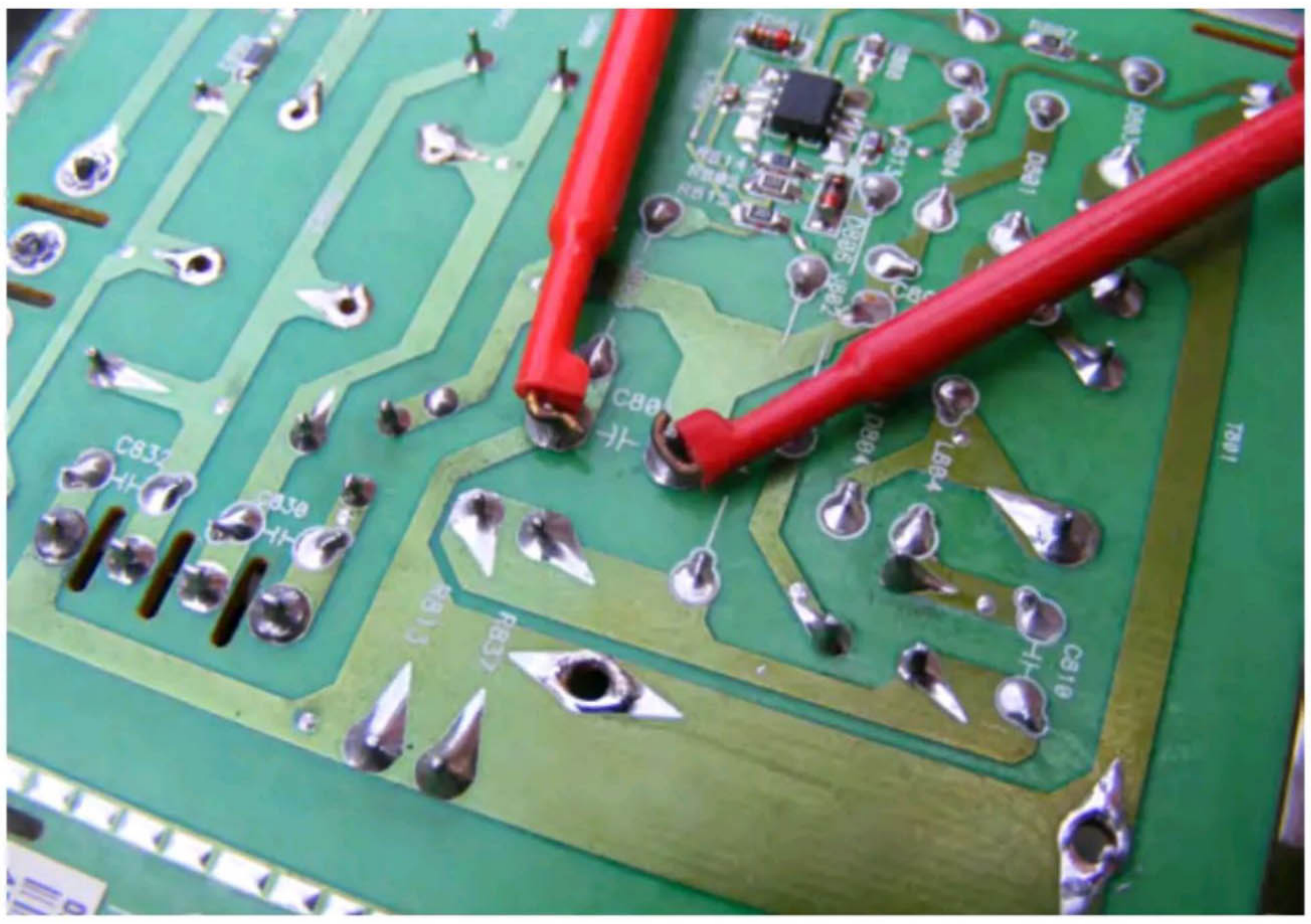


Fig.3: The gripping test probes give time for the capacitor to discharge while leaving your hands free for checking the voltage.

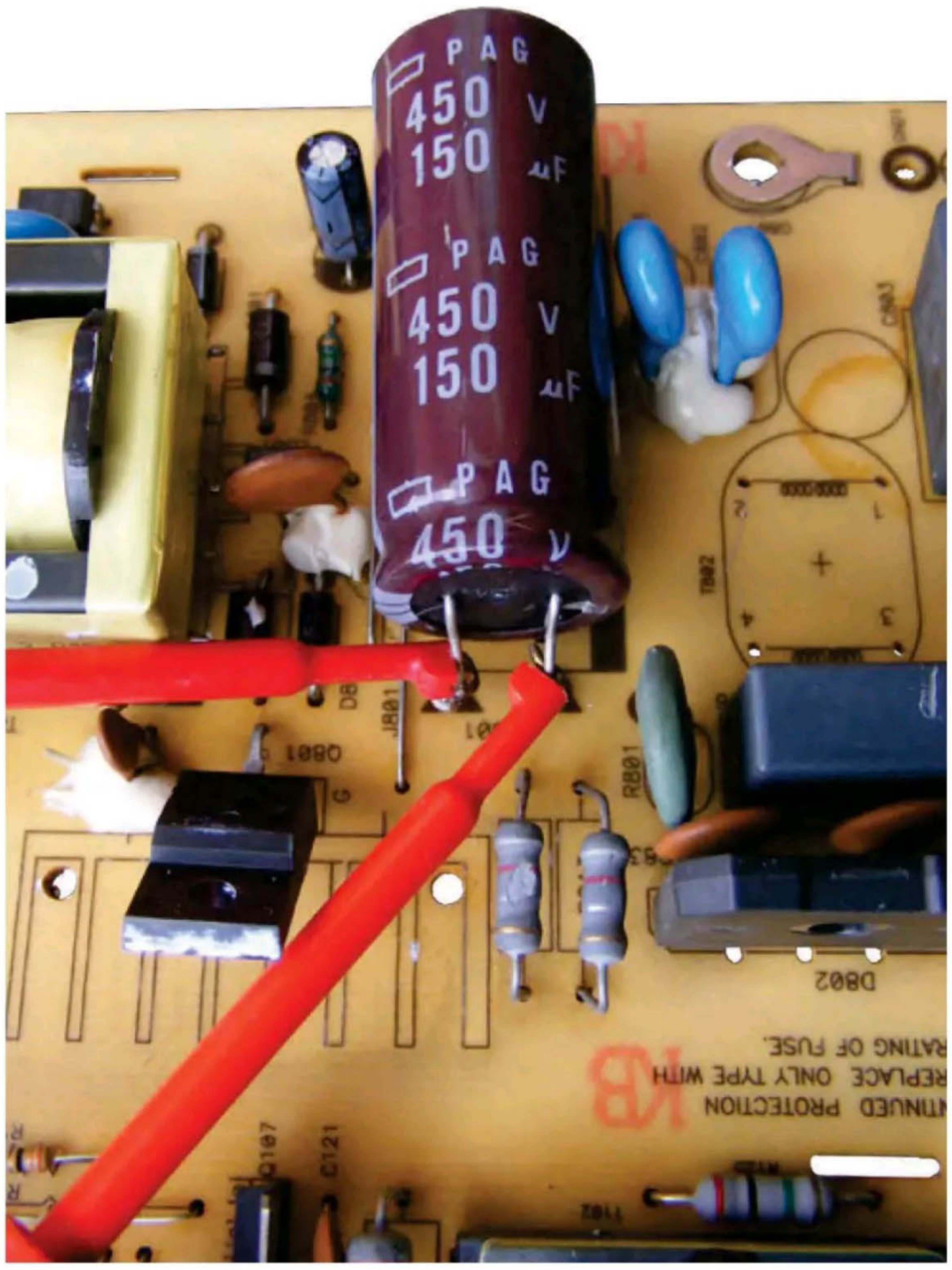


Fig.4: on this Brother printer power supply, the probes can be clipped onto the capacitor leads directly.

GET THE LATEST COPY OF OUR TEACH-IN SERIES AVAILABLE NOW!

ON SALE in WHSmith and other independent newsagents

PRICE £8.99

(includes P&P to UK if ordered direct from us)

FREE CD-ROM
ELECTRONICS TEACH-IN 9
£8.99
GET TESTING!
PIC n' Mix
Teach-In 9
Get Testing

Teach-In 9 – Get Testing!

This series of articles provides a broad-based introduction to choosing and using a wide range of test gear, how to get the best out of each item and the pitfalls to avoid. It provides hints and tips on using, and – just as importantly – interpreting the results that you get. The series deals with familiar test gear as well as equipment designed for more specialised applications.

The articles have been designed to have the broadest possible appeal and are applicable to all branches of electronics. The series crosses the boundaries of analogue and digital electronics with applications that span the full range of electronics – from a single-stage transistor amplifier to the most sophisticated microcontroller system. There really is something for everyone!

Each part includes a simple but useful practical test gear project that will build into a handy gadget that will either extend the features, ranges and usability of an existing item of test equipment or that will serve as a stand-alone instrument. We've kept the cost of these projects as low as possible, and most of them can be built for less than £10 (including components, enclosure and circuit board).

FREE COVER-MOUNTED CD-ROM

On the free cover-mounted CD-ROM you will find the software for the PIC n' Mix series of articles. Plus the full Teach-In 2 book – Using PIC Microcontrollers – A practical introduction – in PDF format. Also included are Microchip's MPLAB ICD 4 In-Circuit Debugger User's Guide; MPLAB PICkit 4 In-Circuit Debugger Quick Start Guide; MPLAB PICkit4 Debugger User's Guide.

ORDER YOUR COPY TODAY JUST CALL 01202 880299
OR VISIT www.electronpublishing.com

DOWNLOAD YOUR PE/EPE BACK ISSUES

COMPLETE-YEAR collections – all 12 issues for each year!

From just £5.95 each

VOL 28 – January to December 1999 – **£5.95**
VOL 29 – January to December 2000 – **£6.95**
VOL 30 – January to December 2001 – **£7.95**
VOL 31 – January to December 2002 – **£7.95**
VOL 32 – January to December 2003 – **£7.95**
VOL 33 – January to December 2004 – **£7.95**
VOL 34 – January to December 2005 – **£8.95**
VOL 35 – January to December 2006 – **£8.95**
VOL 36 – January to December 2007 – **£8.95**
VOL 37 – January to December 2008 – **£8.95**
VOL 38 – January to December 2009 – **£10.95**
VOL 39 – January to December 2010 – **£11.95**
VOL 40 – January to December 2011 – **£12.95**
VOL 41 – January to December 2012 – **£14.95**
VOL 42 – January to December 2013 – **£15.95**
VOL 43 – January to December 2014 – **£16.95**
VOL 44 – January to December 2015 – **£17.95**
VOL 45 – January to December 2016 – **£19.95**
VOL 46 – January to December 2017 – **£22.95**
VOL 47 – January to December 2018 – **£27.95**
VOL 48 – January to December 2019 – **£27.95**
VOL 49 – January to December 2020 – **£29.95**
VOL 50 – January to December 2021 – **£32.95**
VOL 51 – January to December 2022 – **£34.95 – NEW!**

FIVE-YEAR collections – 60 issues!

From just £14.95 each

PE2832: Jan 99 to Dec 03 – **£14.95**
PE2933: Jan 00 to Dec 04 – **£14.95**
PE3034: Jan 01 to Dec 05 – **£14.95**
PE3135: Jan 02 to Dec 06 – **£14.95**
PE3236: Jan 03 to Dec 07 – **£14.95**
PE3337: Jan 04 to Dec 08 – **£14.95**
PE3438: Jan 05 to Dec 09 – **£19.95**
PE3539: Jan 06 to Dec 10 – **£25.95**
PE3640: Jan 07 to Dec 11 – **£25.95**
PE3741: Jan 08 to Dec 12 – **£25.95**
PE3842: Jan 09 to Dec 13 – **£33.95**
PE3943: Jan 10 to Dec 14 – **£33.95**
PE4044: Jan 11 to Dec 15 – **£33.95**
PE4145: Jan 12 to Dec 16 – **£35.95**
PE4246: Jan 13 to Dec 17 – **£37.95**
PE4347: Jan 14 to Dec 18 – **£38.95**
PE4448: Jan 15 to Dec 19 – **£39.95**
PE4549: Jan 16 to Dec 20 – **£42.95**
PE4650: Jan 17 to Dec 21 – **£44.95 – NEW!**

20 YEARS OF PE!

AMAZING BUNDLE OFFER
Every issue from 1999 – 2018

240 magazines for just

£59.95

PE2832: Jan 99 to Dec 03
PLUS!
PE3337: Jan 04 to Dec 08
PLUS!
PE3842: Jan 09 to Dec 13
PLUS!
PE3447: Jan 14 to Dec 18

Do you have the 15-year bundle?

Upgrade to the new 20-year bundle for just £14.95 with: PE15to20

Download your collections at: www.electronpublishing.com

Max's Cool Beans

By Max the Magnificent

Arduino Bootcamp – Part 23 (yet more on BCD!)

I have received several emails from readers saying how much they've been enjoying our forays into the realm of digital logic design. They were talking about the 2:4 decoder we created out of AND and NOT gates, along with our discussions on ring and Johnson counters that can be implemented using D-type flipflops.

A cunning conundrum

On the software side of things, circa the late 1970s, monthly magazines like Byte often published programming competitions. First, they picked a simple microprocessor like an 8-bit 6502, 8080, or Z80. Next, they set a task like counting all the 1s in an 8-bit word using the smallest number of machine code instructions and/or the least amount of memory for both instructions and data.

Similarly, on the hardware side of things, prior to the introduction of logic synthesis technology in the 1980s, digital logic designers delighted in setting and solving logic problems. One of my favorites is shown in Fig.1. We start with a black box (which means we don't know what's inside) that has three inputs (`A`, `B`, `C`) and three outputs (`NotA`, `NotB`, `NotC`).

We want each output to reflect the logical negation of its corresponding input. That is, if `A` is 0 then `NotA` should be 1 and vice versa. Similarly for `B` and `NotB`, and for `C` and `NotC`.

Our mission is to decide on a set of logic gates inside the box that will implement this function. If we could use any gates we wished, the solution would be easy peasy lemon squeezy: just three NOT gates would be sufficient to do the job.

Unfortunately, for the purposes of this puzzle, we are constrained to having only two NOT gates at our disposal. On the bright side, we can also use as many 2- and 3-input AND and OR gates as we wish.

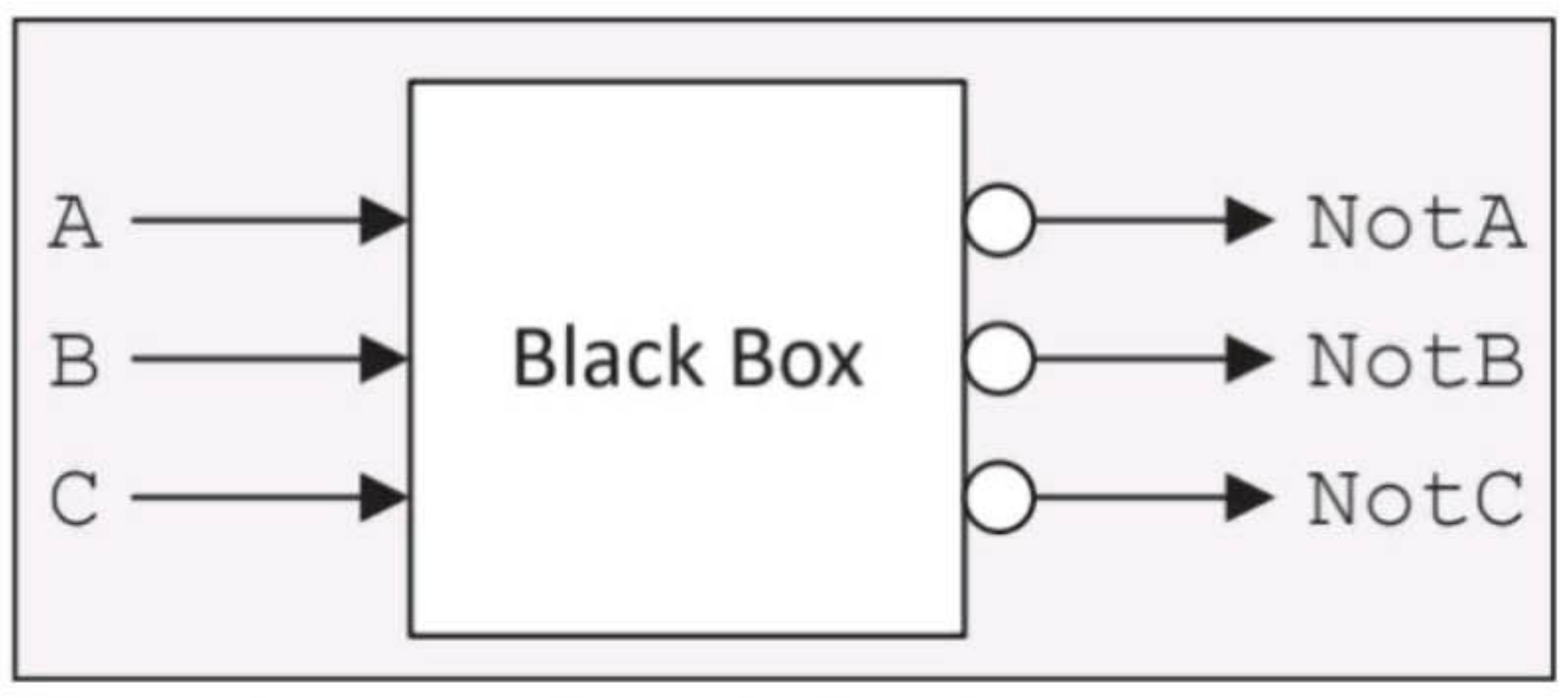


Fig.1: A conundrum indeed.

Even if you've seen this puzzle before (I may have mentioned this little brain-teaser years ago here in *PE*), try to solve it yourself from first principles. However impossible it may seem, there *is* a real-world solution (have I ever lied to you before?).

I will present the solution in next month's column.

Lest we forget

On the off-chance you are new to this series (or mayhap your memory isn't what it used to be), we are in the process of creating a clock using four 7-segment light-emitting diode (LED) displays, a real-time clock (RTC) breakout board (BOB) and an Arduino Uno microcontroller.

We are using this clock as a vehicle to learn interesting stuff while investigating different techniques for driving our displays.

The current state of play is that we are employing a single binary-coded decimal (BCD) to 7-segment decoder in the form of a 7448 integrated circuit (IC) to drive all four of our 7-segment displays.

We've also implemented a home-grown 2-to-4 decoder using a 7408 IC containing four 2-input AND gates and a 7404 IC containing six NOT gates (of which we are only using two).

We are using our decoder to multiplex the displays. That means we are presenting different values to each display, switching them on and off so quickly that our eyes and brains are tricked into thinking that all the displays are lit all the time.

As usual, you can download a PDF of our latest and greatest breadboard, component, and wiring ensemble (with a file name of **CB-Nov24-brd-01.pdf**). As is typical, all the files mentioned in this column are available from the November 2024 page of the PE website: **https://pemag.au/link/ac15**

Data sheet induced disorientation

A top view of our BCD decoder IC is shown in Fig.2. The BCD inputs (`ABCD`) are driven by the Arduino, while the decoded outputs (`abcdefg`) are used to drive the 7-segment displays (via current-limiting resistors).

In our previous column (*PE*, October 2024), we showed the signal names associated with pins 3, 4, and 5 as ? ('don't know') characters. We used a 10kΩ pull-up resistor to place them in their inactive states, and we omitted them from our truth table and discussions.

Well, now it's time to pull aside the curtains to reveal that the signals associated with pins 3, 4, and 5 are called $\overline{\texttt{LT}}$, $\overline{\texttt{BI}}$/$\overline{\texttt{RBO}}$ and $\overline{\texttt{RBI}}$, respectively.

An overscore, overline, or overbar, is a horizontal line immediately above the text. Digital logic designers see these overscores as indicating active-low signals (ie, they are 'on' when low and 'off' when high), which is why we used pull-up resistors to place these pins in their inactive-high states.

That's in contrast to the more intuitive 'active-high' scheme, where 'on' is represented by a high level, and 'off' a low level.

To be honest, we aren't going to use these pins ourselves because we have other *poisson à frire* (fish to fry). Having said that, it's still worthwhile for us to take a moment to consider what they do and how they work their magic.

If we revisit the data sheet for the 7448 (**https://pemag.au/link/abzk**), we find a cornucopia of confusing truth

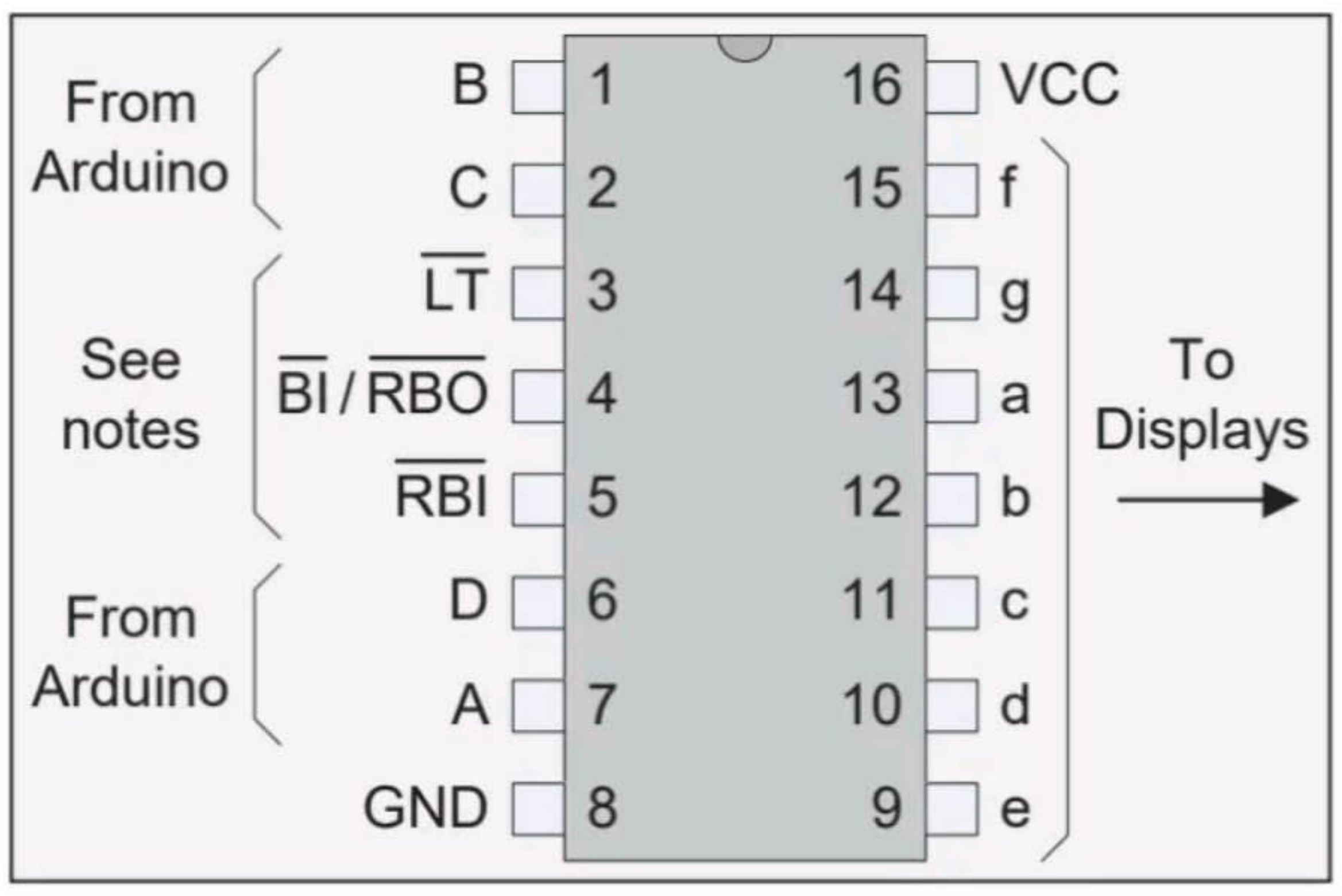


Fig.2: A top view of the 7448 BCD decoder.

Decimal or Function	Inputs						$\overline{\mathrm{BI}}$/$\overline{\mathrm{RBO}}$	Outputs						
	$\overline{\mathrm{LT}}$	$\overline{\mathrm{RBI}}$	D	C	B	A		a	b	c	d	e	f	g
0	H	H	L	L	L	L	H	H	H	H	H	H	H	L
1	H	X	L	L	L	H	H	L	H	H	L	L	L	L
2	H	X	L	L	H	L	H	H	H	L	H	H	L	H
3	H	X	L	L	H	H	H	H	H	H	H	L	L	H
4	H	X	L	H	L	L	H	L	H	H	L	L	H	H
5	H	X	L	H	L	H	H	H	L	H	H	L	H	H
6	H	X	L	H	H	L	H	L	L	H	H	H	H	H
7	H	X	L	H	H	H	H	H	H	H	L	L	L	L
8	H	X	H	L	L	L	H	H	H	H	H	H	H	H
9	H	X	H	L	L	H	H	H	H	H	L	L	H	H
10	H	X	H	L	H	L	H	L	L	L	H	H	L	H
11	H	X	H	L	H	H	H	L	L	H	H	L	L	H
12	H	X	H	H	L	L	H	L	H	L	L	L	H	H
13	H	X	H	H	L	H	H	H	L	L	H	L	H	H
14	H	X	H	H	H	L	H	L	L	L	H	H	H	H
15	H	X	H	H	H	H	H	L	L	L	L	L	L	L
$\overline{\mathrm{BI}}$	X	X	X	X	X	X	L	L	L	L	L	L	L	L
$\overline{\mathrm{RBI}}$	H	L	L	L	L	L	L	L	L	L	L	L	L	L
$\overline{\mathrm{LT}}$	L	X	X	X	X	X	H	H	H	H	H	H	H	H

Fig.3: the complete 74LS48 BCD decoder truth table.

tables. The one we are interested in is Table T2, which I've replicated in Fig.3 for your delight.

Remember that this is an old data sheet that originated in the early 1970s. The L (low) and H (high) values correspond to what, in today's truth tables, we would show as 0s and 1s, respectively. Also, the X characters indicate 'don't care' values. (In the discussions below, I'm going to use 0s and 1s instead of Ls and Hs.)

Wrapping our minds around this truth table can make our brains wobble on their gimbals, so we shall take things step-by-step.

Let's start with the fact that the columns associated with the $\overline{\mathrm{LT}}$ ("lamp test") and $\overline{\mathrm{RBI}}$ ("ripple blanking input") signals are shown on the inputs side of the table.

In contrast, the column associated with the $\overline{\mathrm{BI}}$/$\overline{\mathrm{RBO}}$ ("blanking input"/"ripple blanking output") signal appears *between* the inputs and outputs in the table, because it can act as either an input *or* an output. It manages that trick using something called 'wired-AND' logic, but we don't need to worry about that here.

The reason $\overline{\mathrm{LT}}$ stands for "lamp test" is that, in addition to LEDs, devices of this ilk were originally used to control displays that employed small incandescent bulbs (aka lamps).

For the first 16 rows of the table (indicated as `0` to `15` in the left-hand "Decimal or Function" column), as long as the $\overline{\mathrm{LT}}$ and $\overline{\mathrm{BI}}$/$\overline{\mathrm{RBO}}$ signals are pulled high or remain unconnected (they have weak internal pull-up resistors), we don't care what the $\overline{\mathrm{RBI}}$ signal is doing and the outputs will be decoded versions of the BCD inputs.

Actually, that's not 100% true. Observe the first row (`0` in the left-hand column). In this one case, the $\overline{\mathrm{RBI}}$ signal must be pulled high or left floating (aka unconnected).

In a moment, we shall see why this is the case.

Now let's consider the last three rows (the "Function" rows) in the table, starting with the $\overline{\mathrm{BI}}$ row. This tells us that if we are using the $\overline{\mathrm{BI}}$/$\overline{\mathrm{RBO}}$ signal as an input and we pull it down to its active-low (0) state, we don't care what any of the other inputs are doing because all the outputs driving the display will be set to 0.

Since the 7448 is intended to drive common-cathode displays, this means all the segments on the display will be off, resulting in a 'blank' character, which is why this incarnation of the signal is called the "blanking input".

Next, let's turn our attention to the $\overline{\mathrm{RBI}}$ row. In this case, we are considering the $\overline{\mathrm{BI}}$/$\overline{\mathrm{RBO}}$ signal in its role as an output. This row tells us that as long at the $\overline{\mathrm{LT}}$ input is 1, then when the $\overline{\mathrm{RBI}}$ input is 0 *and* the `ABCD` inputs are 0, this will cause the $\overline{\mathrm{BI}}$/$\overline{\mathrm{RBO}}$ signal and all the outputs driving the display to be set to 0 (our 'blank' character).

In turn, this explains why the $\overline{\mathrm{RBI}}$ signal had to be pulled high or left floating in the first row of the table. It's only when this signal is 0 *and* the `ABCD` inputs are 0 that the outputs will be set to 0; if $\overline{\mathrm{RBI}}$ is 1 when the `ABCD` inputs are 0, the outputs will cause the display to present a "0" character.

The final row returns to assuming that we are using the $\overline{\mathrm{BI}}$/$\overline{\mathrm{RBO}}$ signal as an input that is being pulled high or left floating. In this case, placing the $\overline{\mathrm{LT}}$ signal in its active-low state will cause all the outputs driving the display to be set to 1, which means all the segments on the display will be switched on.

Wow! Look at that!

I don't know about you, but fighting my way through the discussions above made my head hurt. If you want to know how I currently feel, try explaining all this to someone else.

So, why would the designers of the 7448 go to all this trouble? Well, let's try to look at things from their point of view.

The power, ground, four BCD inputs, and seven decoded outputs come to a total of 13 pins. If they had used a 14-pin package, which was very common at that time, this would have left them with only one free pin to play with.

They could have used this pin to implement something like a $\overline{\mathrm{LT}}$ ("lamp test") or $\overline{\mathrm{BI}}$ ("blanking input") function but not both. Instead, they opted to use a 16-pin package, which was also very common, and which left them with three unused pins to play with.

Part of the deliberations surrounding the creation of the 7448 would have involved looking at the problems design engineers (the users of these devices) were trying to solve at that time.

In many cases, those engineers wanted to create pieces of equipment like frequency counters that boasted a bunch of 7-segment displays. For the purposes of these discussions, we will assume those displays had only five digits, but there could easily be 10 or more.

Now suppose we have a value like 42. On a 5-digit display, if we didn't blank the leading zeros, it would be presented as 00042. In addition to being aesthetically unpleasing, this degrades the user's ability to quickly comprehend the value being presented.

As a result, in many cases, we wish to blank any leading zeros. As we've seen with our own clock project, this is relatively easy to do if we are controlling our displays using a microcontroller.

However, back in the early 1970s when the 7448 was introduced, microcontrollers and microprocessors were only just starting to appear on the scene, and very few engineers were using such components in their designs.

Bearing all this in mind, consider a generic 5-digit display using five 7448s BCD decoders. If all the $\overline{\mathrm{RBI}}$ inputs to the decoders are pulled up to their inactive-high states, the result will be no blanking of leading zeros (Fig.4a).

By comparison, consider the case

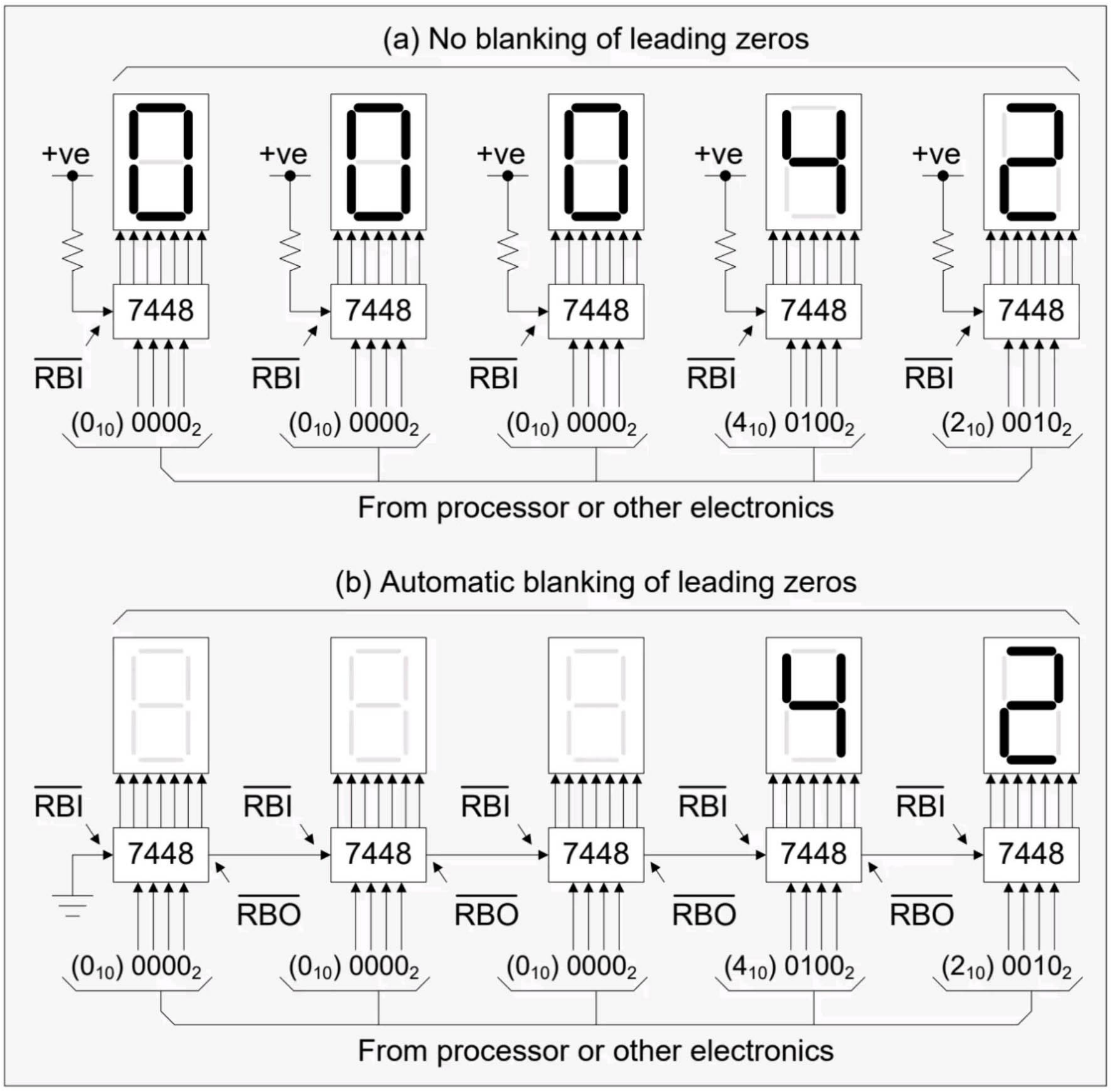

(a) No blanking of leading zeros
+ve
7448
$\overline{\text{RBI}}$
$(0_{10})\ 0000_2$
$(0_{10})\ 0000_2$
$(0_{10})\ 0000_2$
$(4_{10})\ 0100_2$
$(2_{10})\ 0010_2$
From processor or other electronics
(b) Automatic blanking of leading zeros
$\overline{\text{RBI}}$
7448
$\overline{\text{RBO}}$
$(0_{10})\ 0000_2$
$(0_{10})\ 0000_2$
$(0_{10})\ 0000_2$
$(4_{10})\ 0100_2$
$(2_{10})\ 0010_2$
From processor or other electronics

Fig.4: This shows how to use the 7448 feature that provides automatic blanking of leading zeros.

where the $\overline{\text{RBI}}$ input to the most-significant (left-most) BCD decoder is pulled down to its active-low state, after which the $\overline{\text{RBI}}$ inputs to any downstream decoders are fed from the $\overline{\text{RBO}}$ outputs from their upstream counterparts (Fig.4b).

In this case, whenever a decoder sees a 0 on its $\overline{\text{RBI}}$ input *and* 0000 on its ABCD inputs, it will set its $\overline{\text{RBO}}$ output to 0 and disable the outputs driving its 7-segment display.

However, any value other than 0000 on its ABCD inputs will 'break the chain'. That is, this decoder's $\overline{\text{RBO}}$ output will remain at 1, thereby resulting in any downstream decoders seeing 1s on their $\overline{\text{RBI}}$ inputs and displaying any values presented to them, including zeros.

I don't know about you, but I think this is incredibly clever, especially when we consider that all this functionality was realized using less than 40 primitive logic gates, as seen in the logic diagram provided in the data sheet and presented here as Fig.5.

Suffice it to say that I'm constantly amazed by the ingenuity of the design engineers of yesteryear who managed to implement sophisticated functionality using minimal resources.

Meet the CD4511

There are a variety of BCD decoders on the market but a very common type is the CD4511. This part number may be followed by a series of suffixes, but it's anyone's guess as to what they mean.

I asked my friend Joe, who usually emulates a walking encyclopedia when it comes to this sort of thing, but Joe didn't know, so he asked his friend ChatGPT, which responded as follows (I've edited this for size):

'B' usually indicates a buffered version of the device, 'N' usually denotes a standard DIP (dual in-line package) part, 'P' could indicate a plastic DIP, 'C' might be connected to a commercial temperature range, and 'I' might indicate an industrial temperature range.

I must say that these "usually", "could", and "might" qualifiers make that information a little less helpful than I would have liked.

What the heck. I found a pack of 20 CD4511BE devices on Amazon for around £14 (https://pemag.au/link/ac11), which seems like a good deal to me.

I have no clue what the 'E' suffix might be trying to tell us, but at least nothing has Exploded… yet.

If we look at the data sheet (https://pemag.au/link/ac12), we discover something very interesting.

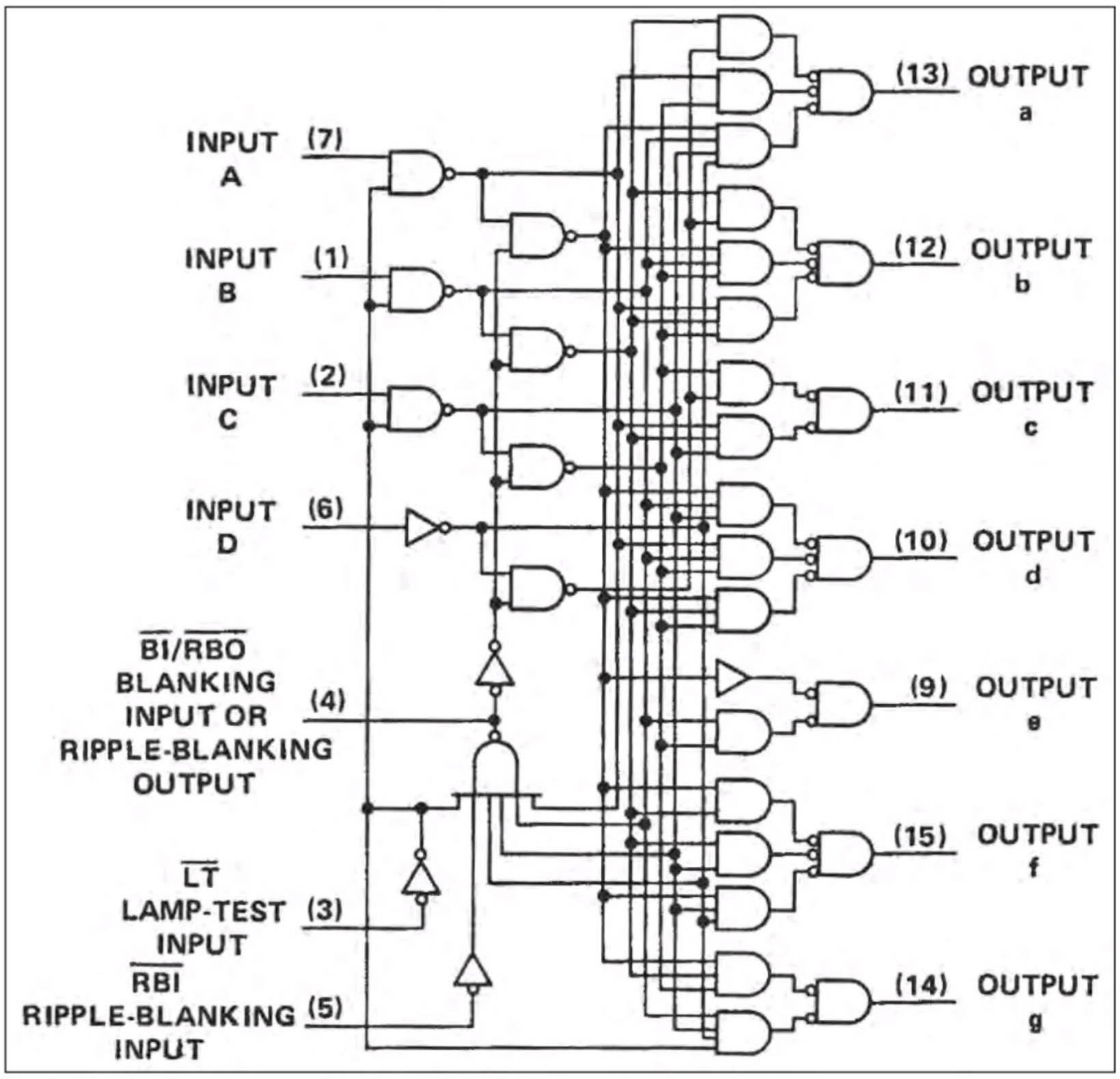

INPUT A (7)
INPUT B (1)
INPUT C (2)
INPUT D (6)
$\overline{\text{BI}}/\overline{\text{RBO}}$ BLANKING INPUT OR RIPPLE-BLANKING OUTPUT (4)
$\overline{\text{LT}}$ LAMP-TEST INPUT (3)
$\overline{\text{RBI}}$ RIPPLE-BLANKING INPUT (5)
(13) OUTPUT a
(12) OUTPUT b
(11) OUTPUT c
(10) OUTPUT d
(9) OUTPUT e
(15) OUTPUT f
(14) OUTPUT g

Fig.5: The logic gates used in the 74LS48 IC, from its data sheet.

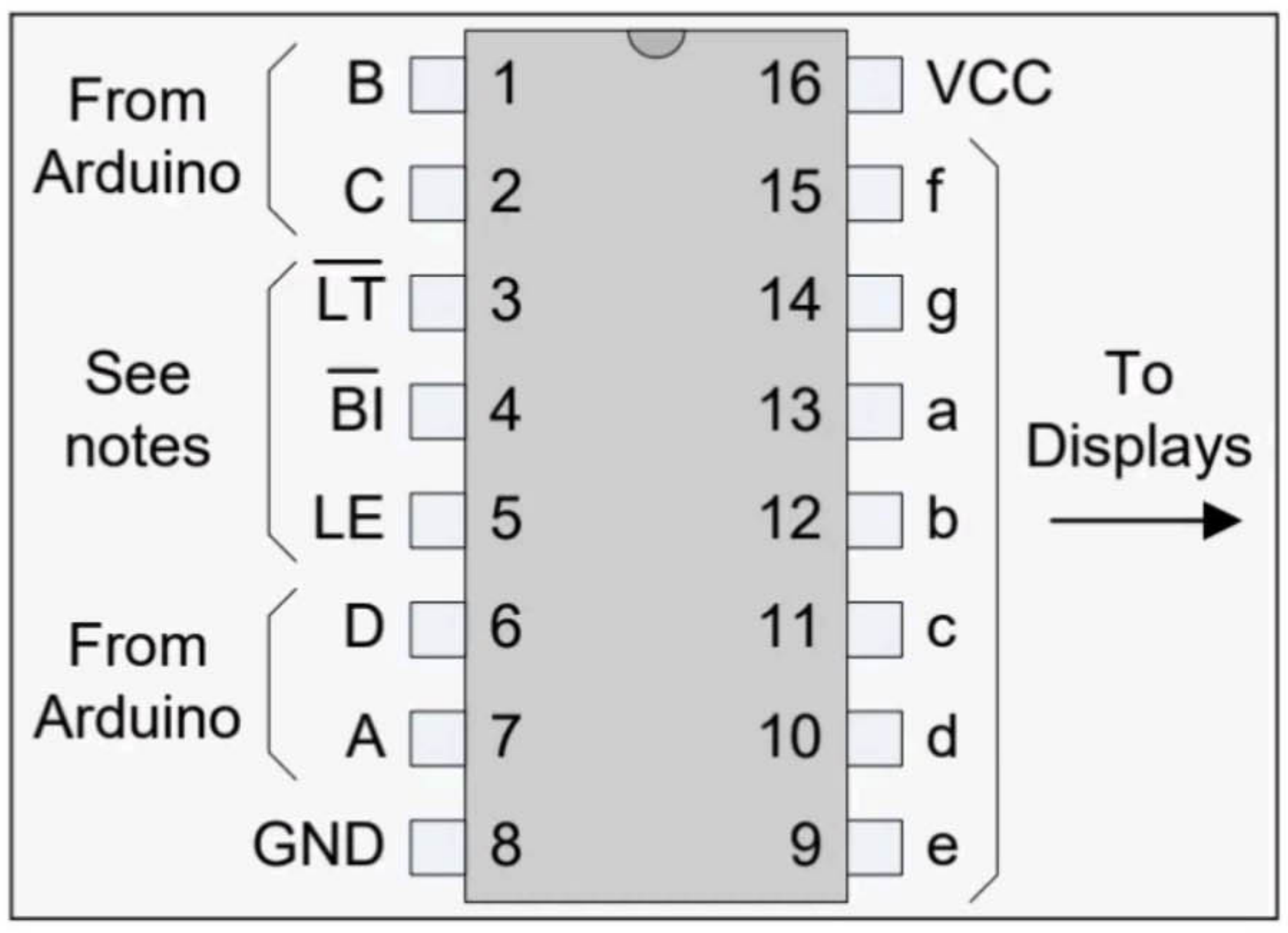


Fig.6: A top view of the CD4511 IC.

Consider the top view of this device, shown in Fig.6.

Observe that, except for two of its three control pins, this little rascal is pin-compatible with the 7448 we've been using. As we see, pin 3 still provides the $\overline{\texttt{LT}}$ ("lamp test") function to turn all the segments on, and pin 4 also still offers the $\overline{\texttt{BI}}$ ("blanking input") function to turn all the segments off, thereby allowing us to show a blank character. However, pin 4 no longer implements the $\overline{\texttt{RBO}}$ ("ripple blanking output") function.

The most obvious difference is exhibited by pin 5. As opposed to performing the role of the 7448's $\overline{\texttt{RBI}}$ ("ripple blanking input"), this pin now acts as the CD4511's active-high `LE` ("latch enable") input.

What does this pin do? Well, let's ponder the CD4511's truth table (Fig.7), along with the segment identification, numerical designations and resultant displays (Fig.8), where all is revealed.

From the first row in the table, we see that when the $\overline{\texttt{LT}}$ input is in its active-low state, we don't care what values are on the other inputs; all the outputs will be on. Similarly, from the second row, we see that when the $\overline{\texttt{BI}}$ input is in its active-low state, so long as the $\overline{\texttt{LT}}$ input is in its inactive-high state, we don't care what values are on the other inputs; all the outputs will be off.

When both the $\overline{\texttt{LT}}$ and $\overline{\texttt{BI}}$ inputs are in their inactive-high states, and the `LE` input is set to 0, whatever BCD values are presented to the `ABCD` inputs will be decoded and the result presented on the `abcdefg` outputs. The BCD codes 0000 through 1001 result in displayed characters of '0' through '9', respectively.

Unlike the designers of the 7448, the creators of the CD4511 took the time and effort to decode the unused input values of 1010 through 1111 into blank characters.

It's the table's last row that calls for our attention. Once the `LE` input has transitioned from a 0 to a 1, whatever values were present on the outputs prior to this transition are latched (locked). After this, any future changes to the `ABCD` inputs will have no effect.

The design of the CD4511 is clever on many levels, not least that, since it was developed after the 7448, its near pin-compatibility with the 7448 meant that product developers could transition between devices with relative ease.

Inputs							Outputs							Display
LE	$\overline{\text{BI}}$	$\overline{\text{LT}}$	D	C	B	A	a	b	c	d	e	f	g	
X	X	0	X	X	X	X	1	1	1	1	1	1	1	All On
X	0	1	X	X	X	X	0	0	0	0	0	0	0	Blank
0	1	1	0	0	0	0	1	1	1	1	1	1	0	0
0	1	1	0	0	0	1	0	1	1	0	0	0	0	1
0	1	1	0	0	1	0	1	1	0	1	1	0	1	2
0	1	1	0	0	1	1	1	1	1	1	0	0	1	3
0	1	1	0	1	0	0	0	1	1	0	0	1	1	4
0	1	1	0	1	0	1	1	0	1	1	0	1	1	5
0	1	1	0	1	1	0	0	0	1	1	1	1	1	6
0	1	1	0	1	1	1	1	1	1	0	0	0	0	7
0	1	1	1	0	0	0	1	1	1	1	1	1	1	8
0	1	1	1	0	0	1	1	1	1	0	0	1	1	9
0	1	1	1	0	1	0	0	0	0	0	0	0	0	Blank
0	1	1	1	0	1	1	0	0	0	0	0	0	0	Blank
0	1	1	1	1	0	0	0	0	0	0	0	0	0	Blank
0	1	1	1	1	0	1	0	0	0	0	0	0	0	Blank
0	1	1	1	1	1	0	0	0	0	0	0	0	0	Blank
0	1	1	1	1	1	1	0	0	0	0	0	0	0	Blank
1	1	1	X	X	X	X	*	*	*	*	*	*	*	*

*Same as the BCD code applied prior to the 0-to-1 transition on LE

Fig.7: The CD4511 truth table.

Multiplex no more

We could replicate our existing 7448-based clock implementation using a CD4511 with a single set of current-limiting resistors. As before, we could multiplex our four 7-segment displays using a 2-to-4 decoder and four transistors.

However, the fact that the CD4511 provides latching functionality allows us to opt for an alternative approach, as illustrated in Fig.9.

In this case, each 7-segment display is provided with its own set of current-limiting resistors and its own CD4511 BCD decoder. The `ABCD` inputs to all the CD4511s are connected in parallel. The `LE` input to each CD4511 is controlled individually. Our default state will be to have all the LE signals set to 1, which means the CD4511s won't 'see' or respond to any activity on the `ABCD` inputs.

We are no longer multiplexing our displays. All we need to do is place a BCD value on the `ABCD` signals and then apply a negative-going (1→0→1 ⊔) pulse to the `LE` input associated with whichever CD4511 we wish to load the BCD value into.

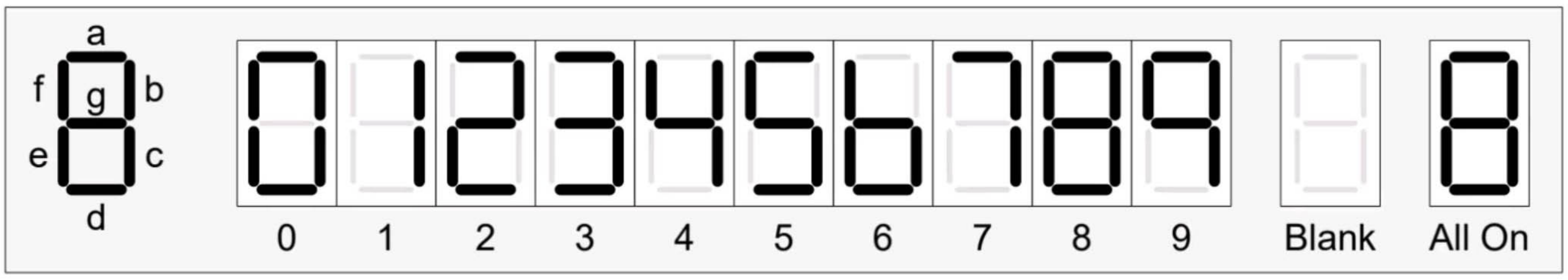
Different approaches have their own advantages and disadvantages. The advantages of the 7448-based multiplexing technique using a 2-to-4 decoder is that it requires one BCD decoder, one set of current-limiting resistors, and (in the case of four displays) only six of the Arduino's pins. The disadvantage is that the more displays we add, the dimmer they get because each display is active for only part of the time.

The advantage of the CD4511-based latching approach is that all the displays are on all the time, which means we can add more displays without any dimming. The disadvantages are that we need a separate BCD decoder with current-limiting resistors for each display and, in the case of four displays, we're using eight of the Arduino's pins (although there are ways to reduce that…).

As one final point of interest, observe that Fig.9 shows the $\overline{\texttt{LT}}$ and $\overline{\texttt{BI}}$ inputs being pulled up to their inactive states. Suppose that – in the fullness of time – we desire the ability to vary the brightness of our displays, perhaps to implement a dimmed nighttime mode.

We could easily implement that function with our CD4511 devices by connecting all their $\overline{\texttt{BI}}$ inputs together and driving them using one of the Arduino's pulse-width modulation

Fig.8: Display segment identification and the possible displays from the CD4511, as per the data sheet.



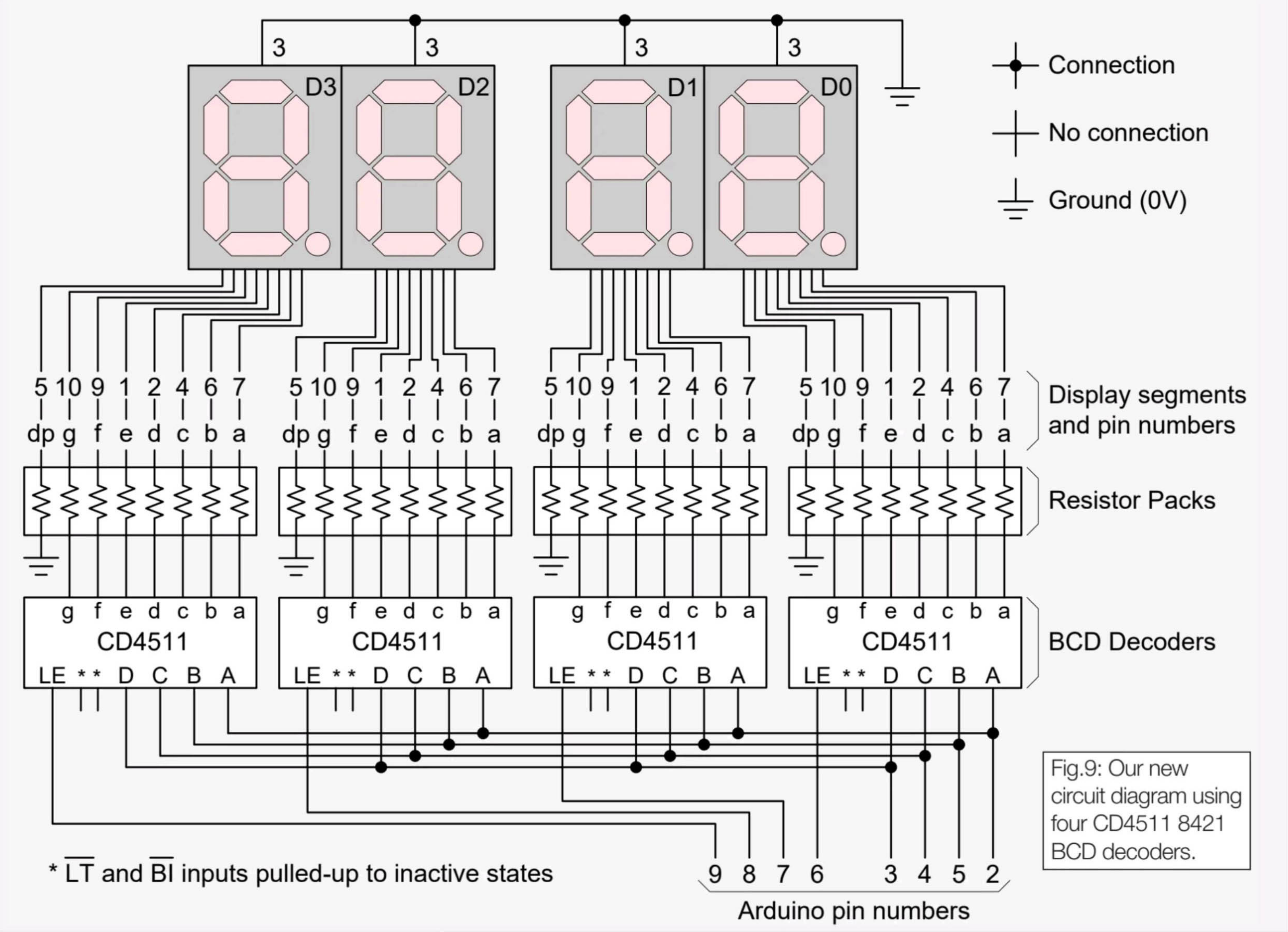

3
D3
D2
D1
D0
Connection
No connection
Ground (0V)
5 10 9 1 2 4 6 7
dp g f e d c b a
Display segments and pin numbers
Resistor Packs
g f e d c b a
CD4511
LE * * D C B A
BCD Decoders
* $\overline{\text{LT}}$ and $\overline{\text{BI}}$ inputs pulled-up to inactive states
9 8 7 6 3 4 5 2
Arduino pin numbers

Fig.9: Our new circuit diagram using four CD4511 8421 BCD decoders.

(PWM)-equipped output pins (we introduced the concept of PWM in *PE*, March 2023).

In this case, we could use the Arduino pin directly; that is, we wouldn't have to employ any transistors (the use of which we introduced in *PE*, January 2024) because the CD4511s handle the current associated with driving their displays.

Preparing to rock and roll

If you have spares of everything, including breadboards, you may decide to keep your existing clock as-is and start a new version. Alternatively, if you've splashed your cash only on an as-needed basis, you will need to remove the components from your current breadboards before proceeding.

The following instructions assume we're building everything from the ground up.

We're going to go through this with our racing sneakers on because we've already done everything discussed here in one form or another. Let's begin by setting up two breadboards as shown in Fig.10 (the full setup can be found in the file named **CB-nov24-brd-02.pdf**).

I've decided to use current-limiting resistors values of 1kΩ in the green and blue power LED assemblies, thereby limiting each LED's current

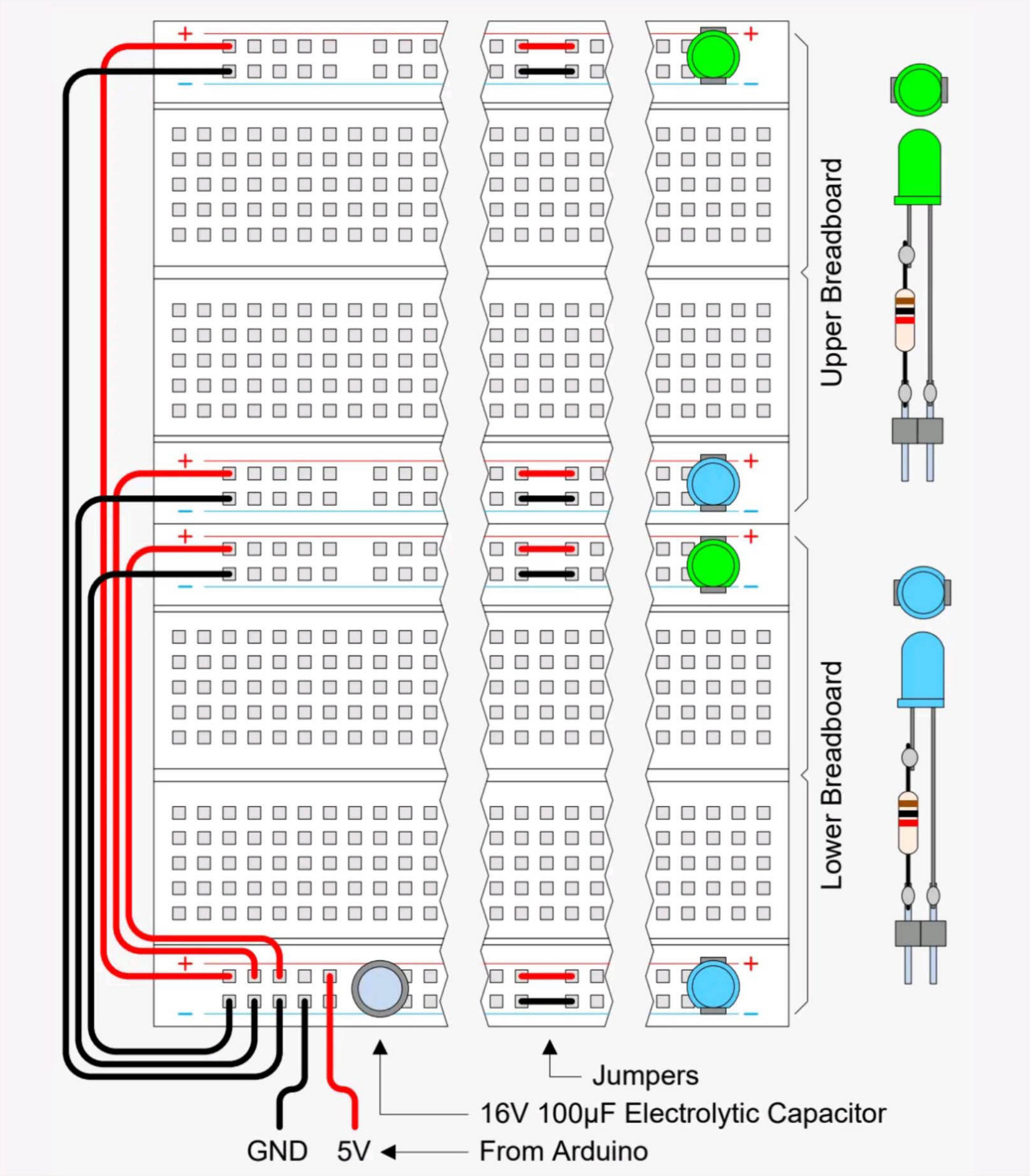

+
−
Upper Breadboard
Lower Breadboard
Jumpers
16V 100µF Electrolytic Capacitor
GND
5V
From Arduino

Fig.10: Our new breadboard setup.

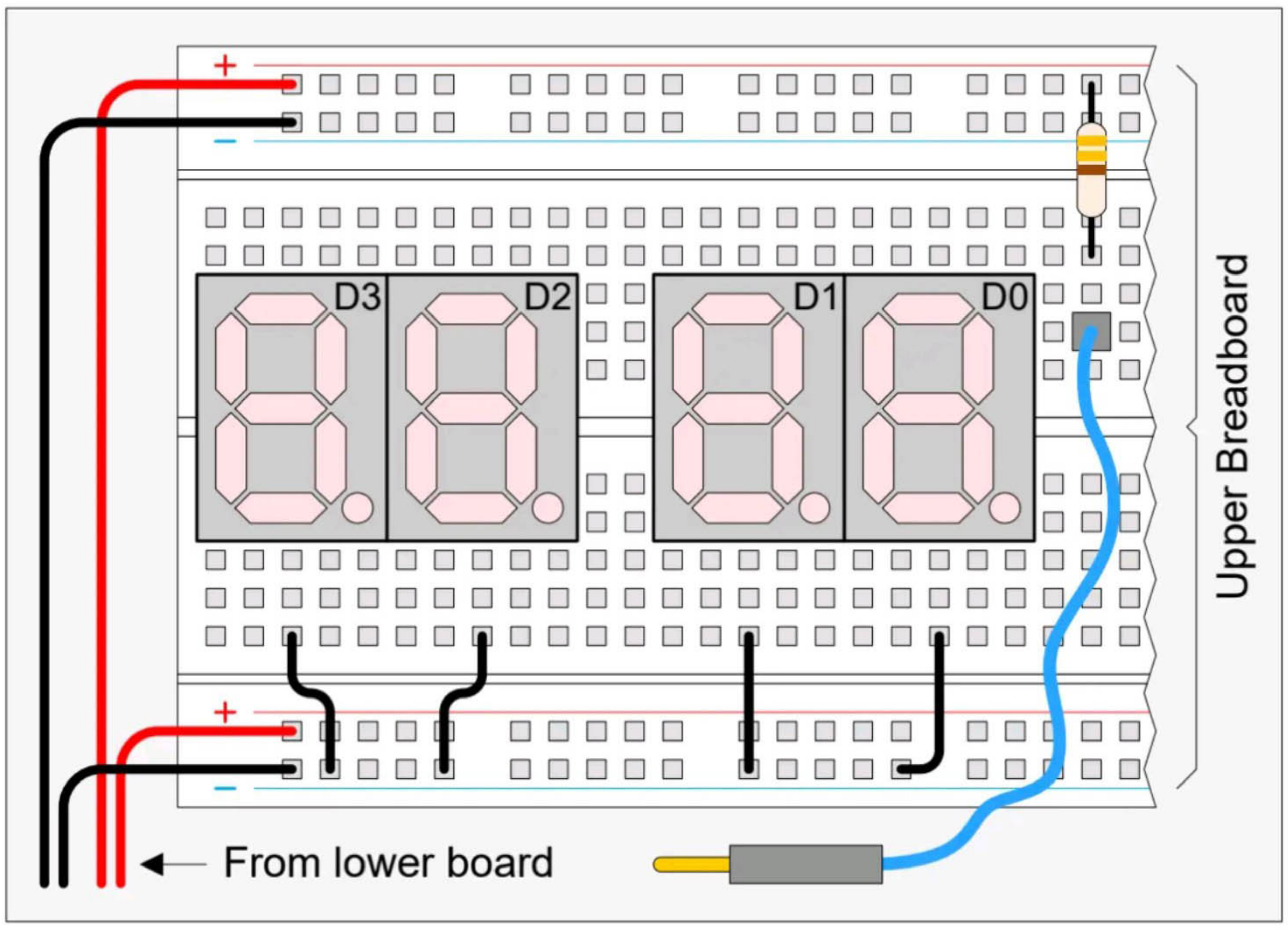


Fig.11: Adding the four 7-segment displays.

consumption to only 2mA.

Once you've wired everything up, plug in the USB cable to power-up your Arduino, then use your multimeter to verify that you are seeing +5V (give or take) across all the power and ground rail pairs.

The easiest way to do this is to apply your multimeter's probes to the component leads soldered to the header pins on the green and blue LED/current-limiting resistor assemblies.

Adding the 7-segment displays

Power everything down and add four 7-segment displays on the left-hand side of the upper breadboard as illustrated in Fig.11.

Make sure to add four wires connecting pin 3 on each display to the nearest ground rail (the full setup can be found in the file named **CB-nov24-brd-03.pdf**).

Before you do anything else, power it up and use your multimeter to re-verify that you are seeing +5V (give or take) across all the power and ground rail pairs.

Observe the temporary current-limiting resistor and flying lead. Remember that we've connected pin 3 of each display to the 0V rail. Also remember that pins 3 and 8 are connected inside each display.

Use the end of the flying lead to verify that the segments associated with pins 1, 2, 4, 5, 6, 7, 9, and 10 of each display function as expected, after which you can power it down and remove the lead and resistor from your setup.

As reflected in Fig.11, I used a 330Ω resistor for this test. Anything between 150Ω and 680Ω would be OK.

The two competing criteria for the resistor's value are (a) to be low enough to light the LED segments sufficiently without burning out the LED and (b) to be high enough to adequately limit the current should we accidentally probe pins 3 or 8, thereby shorting our 5V and 0V rails.

Adding the current-limiting resistors

Our next task is to add the current-limiting resistors to the right-hand side of the upper breadboard. In this case, we are going to use 270Ω resistors (we will consider the rationale behind this value in a moment).

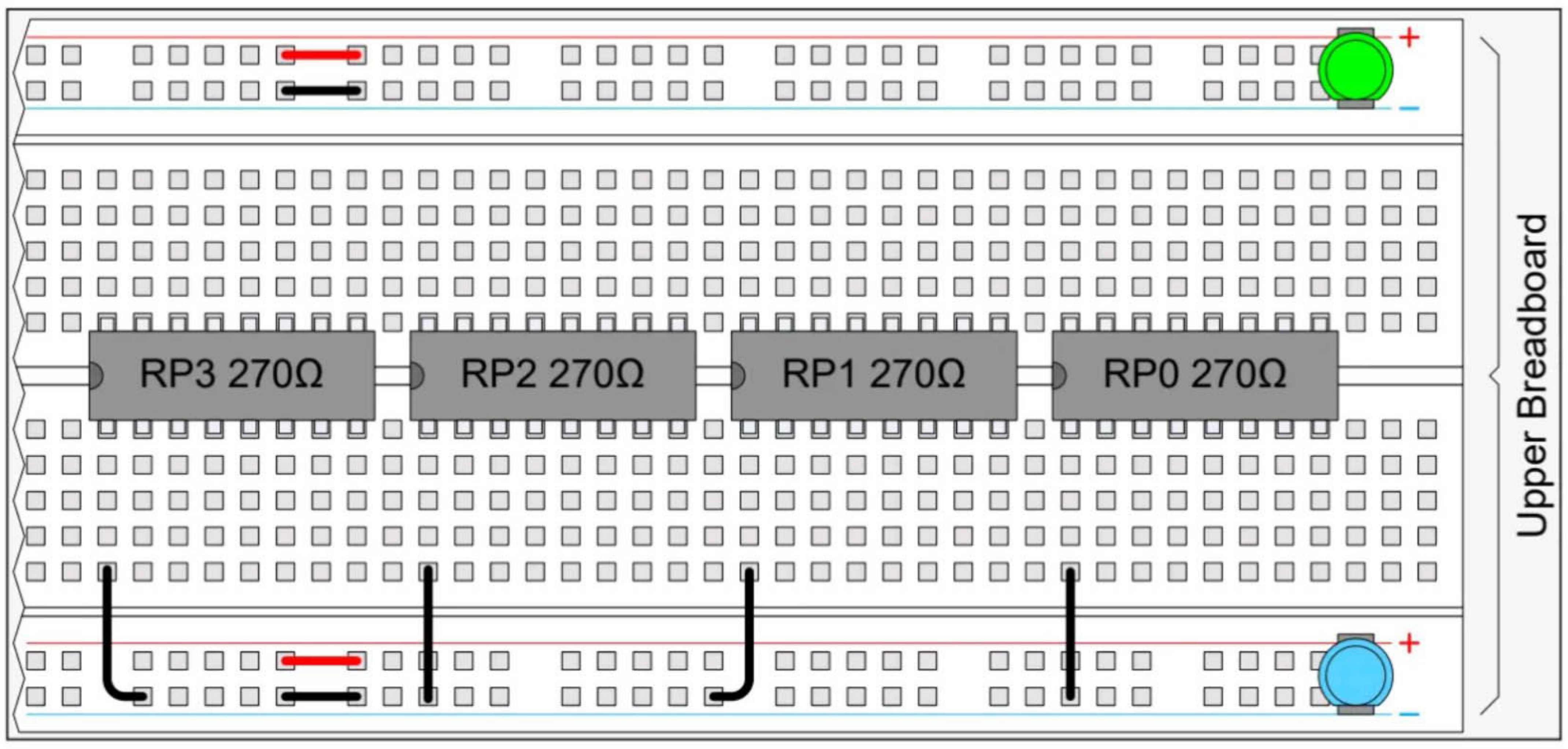


Fig.12: Adding the four resistor networks.

One option is to use 4 × 8 = 32 individual resistors, but that would be a pain. Another possibility is to build four 8-resistor BOBs (we discussed how to build these in *PE*, July 2024).

A third option – the one I went for – is to use four pre-built resistor packs, each presented in a 16-pin package containing eight isolated 270Ω resistors, as illustrated in Fig.12.

These components are offered by various manufacturers, such as the Bourns 4100R Series (https://pemag.au/link/ac13). They are available from most major component vendors, like Digikey (https://pemag.au/link/ac14).

Add wires connecting pin 1 on each resistor pack to the nearest ground rail (the full setup can be found in the file named **CB-nov24-brd-04.pdf**).

We are doing this because the other side of these resistors (pin 16 on the packs) are going to be connected to the decimal point ("dp") segments on the displays. Since we aren't using these segments in this implementation, connecting these resistor pack pins to ground will ensure those segments remain off.

Once again, power everything up and use your multimeter to re-verify that you are seeing +5V (give or take) across all the power and ground rail pairs.

Cunning calculations

So, why did we decide to use 270Ω for our current-limiting resistors? From earlier columns we know that the red LED segments in our displays can support up to 20mA, which we could achieve using 150Ω resistors.

Your first guess might be that it's the current sourcing capability of the CD4511s that's the problem. However, when we look at the CD4511 data sheet, we see it proudly proclaiming: "High current sourcing outputs (up to 25mA)".

The real reason is limitations in our power source. I'm assuming that you, like me, are using a USB cable to power your Arduino Uno, which is, in turn, powering your breadboards. I'm also assuming you are using a USB 2.0 port, which can deliver up to 500mA at 5V, so this is the limiting factor.

I had a quick Google while no one was looking to discover that an Arduino Uno R3 has an average current consumption of 50mA.

As was previously noted, with their 1kΩ current-limiting resistors, each of our four green and blue

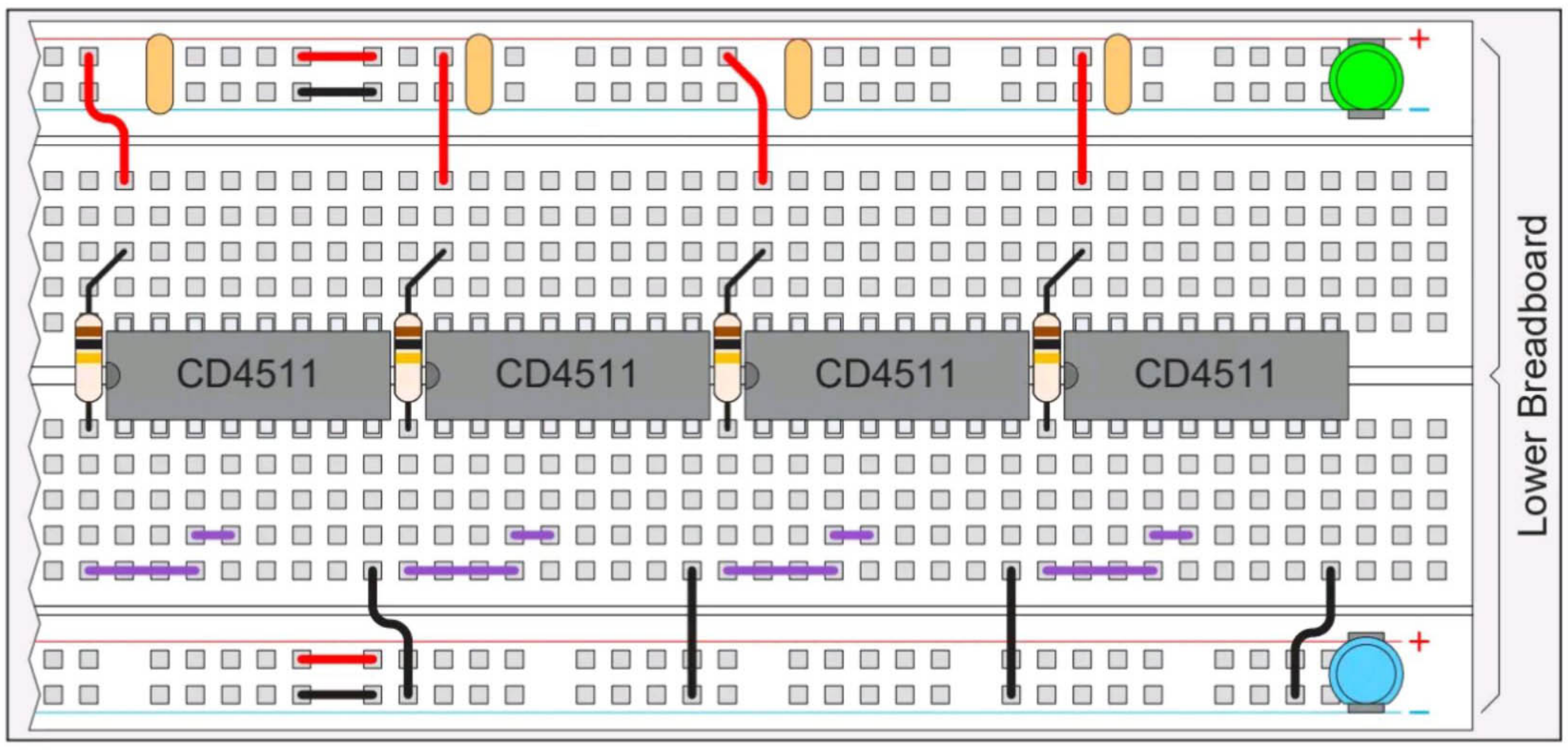


Fig.13: Adding the four BCD decoder chips.

power LED assemblies consumes only 2mA, which equates to 8mA in all. Our DS3231 real-time clock (RTC) consumes a negligible amount of current, but let's assume 1mA to be on the safe side.

"What about our four CD4511 devices?" I hear you cry. Well, the term "quiescent" means "calm", "inactive" or "dormant". In electronics, "quiescent current" refers to the amount of current being consumed by an IC when it is enabled but not doing much.

If we return to our trusty data sheet, we see that the quiescent current for a CD4511 is 20µA at 25°C. This means that, even if we were using five of these devices, their total quiescent current would be only 100µA or 0.1mA. Again, let's say 1mA to be on the safe side.

So, excluding our 7-segment displays, our total current consumption is 60mA. Let's also hold 90mA in reserve to power other functions – like a buzzer, perhaps – in the future, which means we have 500mA – 60mA – 90mA = 350mA remaining to power our displays.

Since we aren't using the "dp" segments on our displays, our worst-case scenario would be for all four displays to be presenting '8' characters (ie, "88 88"), each of which lights seven segments, resulting in 4 × 7 = 28 segments lit.

If we felt like being clever (or if we were desperate), we might note that, in a 24-hour mode, our hours digits will display values of "00" through "23". Alternatively, in a 12-hour mode, they will display values of "01" through "12".

This means that the most-significant hours digit only ever displays "0", "1", or "2", with 6, 2, and 5 segments lit, respectively (referring to Fig.8). Meanwhile, the least-significant hours digit may be used to display "0" through "9", of which "8" lights all seven segments.

Similarly, our minutes digits will display values of "00" through "59". The worst case for the most-significant minutes digit is "0", which lights six segments, while the worst case for the least-significant minutes digit is "8", which lights all seven.

Putting all this together, we could argue that, if we only ever used our displays to implement a clock, our worst-case scenario would occur at eight minutes past eight ("08 08"), which would light 6 + 7 + 6 + 7 = 26 segments.

On the other hand, it's possible to be too clever ("you're so sharp you'll cut yourself", as my dear old grannie used to say). Suppose we decided to use our displays to present other information. In this case, "88 88", lighting 28 segments could be a possibility, so let's go with this.

Based on our self-imposed limit of 350mA, this means each segment has 350mA/28 = 12.5mA at its disposal. Since the red segment LEDs have a forward voltage drop of 2V, we can calculate our ideal current-limiting resistor values as (5V – 2V) ÷ 0.0125A = 240Ω. The closest (and next highest) value in the Bourns series is 270Ω, so that's what I used.

A pert little poser

Why should I do all the work? I have a pert little poser for you to ponder.

Assume our displays are only ever going to be used to implement a clock. Also assume that this clock can support both 12- and 24-hour modes.

Now suppose that, in addition to everything we've discussed above, we decide to suppress any leading zeros in the most-significant hours digit (we introduced this concept in *PE*, August 2024).

In this case, what would be the maximum number of segments we might see active at the same time?

And, based on this, what would be the ideal value for our current-limiting resistors to obtain the maximum possible segment current value, assuming a total allocation of 350mA to power all four of our 7-segment displays?

We will contemplate the conclusion to this conundrum in next month's column. For now, however, onwards and upwards…

Adding the CD4511s

Ensure that everything is powered down, then add the four CD4511 BCD decoders to the right-hand side of the lower breadboard, as shown in Fig.13.

As part of this, add wires connecting pin 8 on each CD4511 to the nearest ground rail, along with wires connecting pin 16 on each IC to the nearest 5V rail. Also add a 0.1µF ceramic capacitor for each IC as shown, along with a 10kΩ pull-up resistor for each.

Once side of each of these resistors should connect to 5V, while the other side connects pins 3 and 4 (the $\overline{\text{LT}}$ and $\overline{\text{BI}}$ inputs) of its respective IC. The latest incarnation of the complete setup can be found in the file named

Online resources

For the purposes of this series, I'm going to assume that you are already familiar with fundamental concepts like voltage, current and resistance. If not, you might want to start by perusing and pondering a short series of articles I penned on these very topics – see: https://pemag.au/link/ac16

Similarly, I'll assume you are no stranger to solderless breadboards. Having said this, even if you've used these little scamps before, there are some aspects to them that can trap the unwary, so may I suggest you feast your orbs on a column I wrote just for you – see: https://pemag.au/link/ac17

Last, but not least, you will find a treasure trove of resources at the **Arduino.cc** website, including example programs and reference documentation.

Cool bean Max Maxfield (Hawaiian shirt, on the right) is emperor of all he surveys at **CliveMaxfield.com** – the go-to site for the latest and greatest in technological geekdom.

Comments or questions? Email Max at: **max@CliveMaxfield.com**

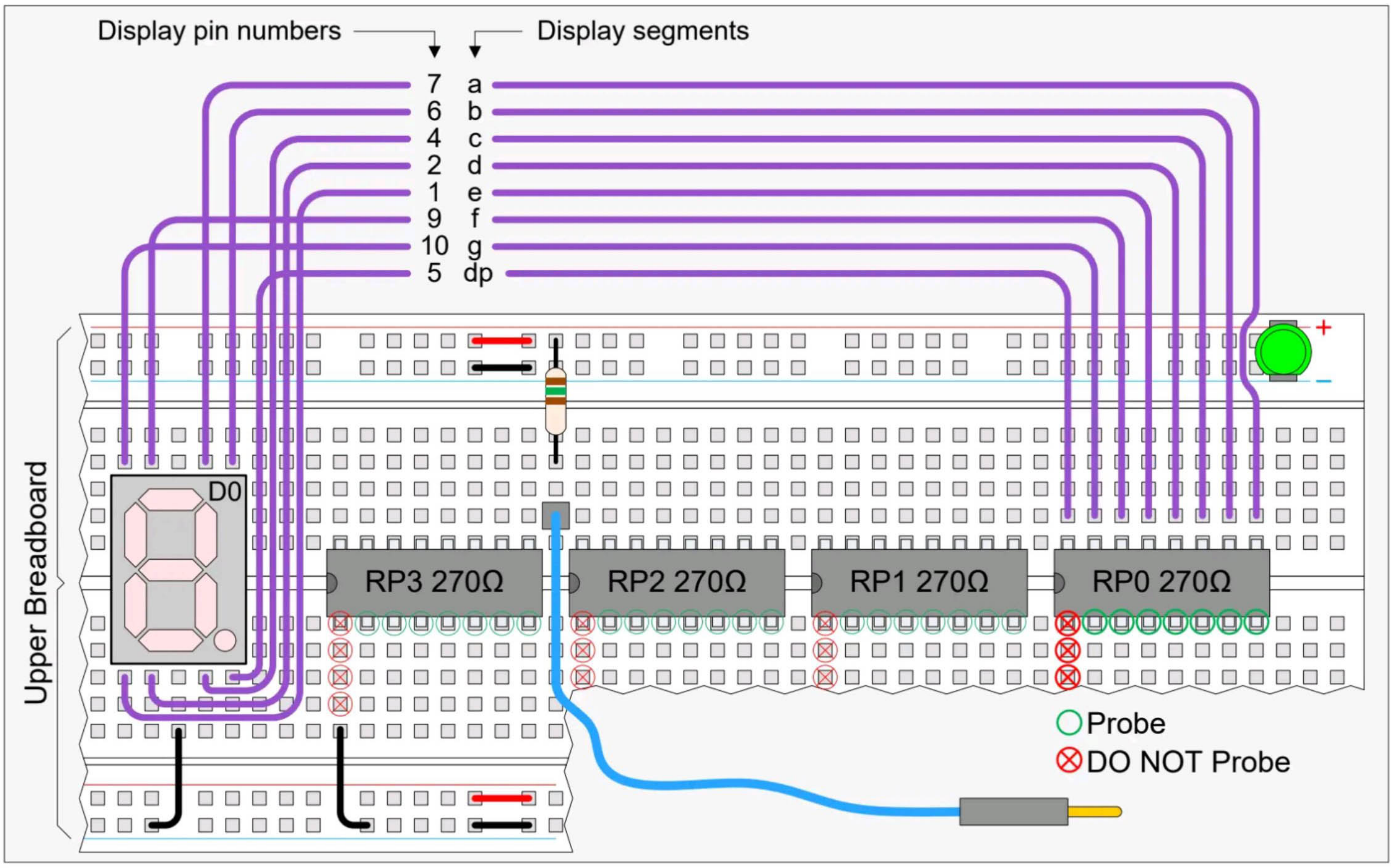

Display pin numbers
Display segments
7 a
6 b
4 c
2 d
1 e
9 f
10 g
5 dp
Upper Breadboard
D0
RP3 270Ω
RP2 270Ω
RP1 270Ω
RP0 270Ω
+
–
Probe
DO NOT Probe

Fig.14: Connecting the first 7-segment display.

CB-nov24-brd-05.pdf.

As usual, power everything up and use your multimeter to re-verify that you are seeing +5V (give or take) across all the power and ground rail pairs. Also, check that you are seeing +5V between pins 8 and 16 on each IC. As well, verify that you are seeing +5V between pin 8 and both of pins 3 & 4 on each IC.

Wiring up the first display

Power everything down, then connect pins 9 through 16 of resistor pack RP0 to display D0 as shown in Fig.14. Also add a temporary current-limiting resistor and flying lead as shown.

Power-up the Arduino, touch the end of the flying lead to pins 8, 7, 6, 5, 4, 3, and 2 of resistor pack RP0, and observe segments a, b, c, d, e, f and g light up on display D0, respectively.

The reason for the current-limiting resistor is to prevent problems if you accidentally touch pin 1 on RP0, thereby shorting the 0V and 5V rails.

Once you've verified the segments, you can power everything down and remove the lead and resistor from your setup (you can see a full drawing in the file named **CB-nov24-brd-06.pdf**).

Connecting the Arduino

Until now, we've been using only the 0V (GND) and +5V pins from the Arduino. Now it's time to connect the digital outputs driving our clock as shown in Fig.15 (a full version of the setup is available in the file named **CB-nov24-brd-07.pdf**).

Observe the BCD and `LE` probe points. As you may recall, we discussed the concept of logic probes in a previous issue (*PE*, August 2024) If you haven't already done so, use a half-size breadboard to create four such probes as illustrated in Fig.16.

Connect probes 0, 1, 2, and 3 in Fig.16 to the corresponding `ABCD` BCD probe points shown in Fig.15.

Remember to connect the 0V and 5V wires connecting the logic probe breadboard to the main clock breadboards.

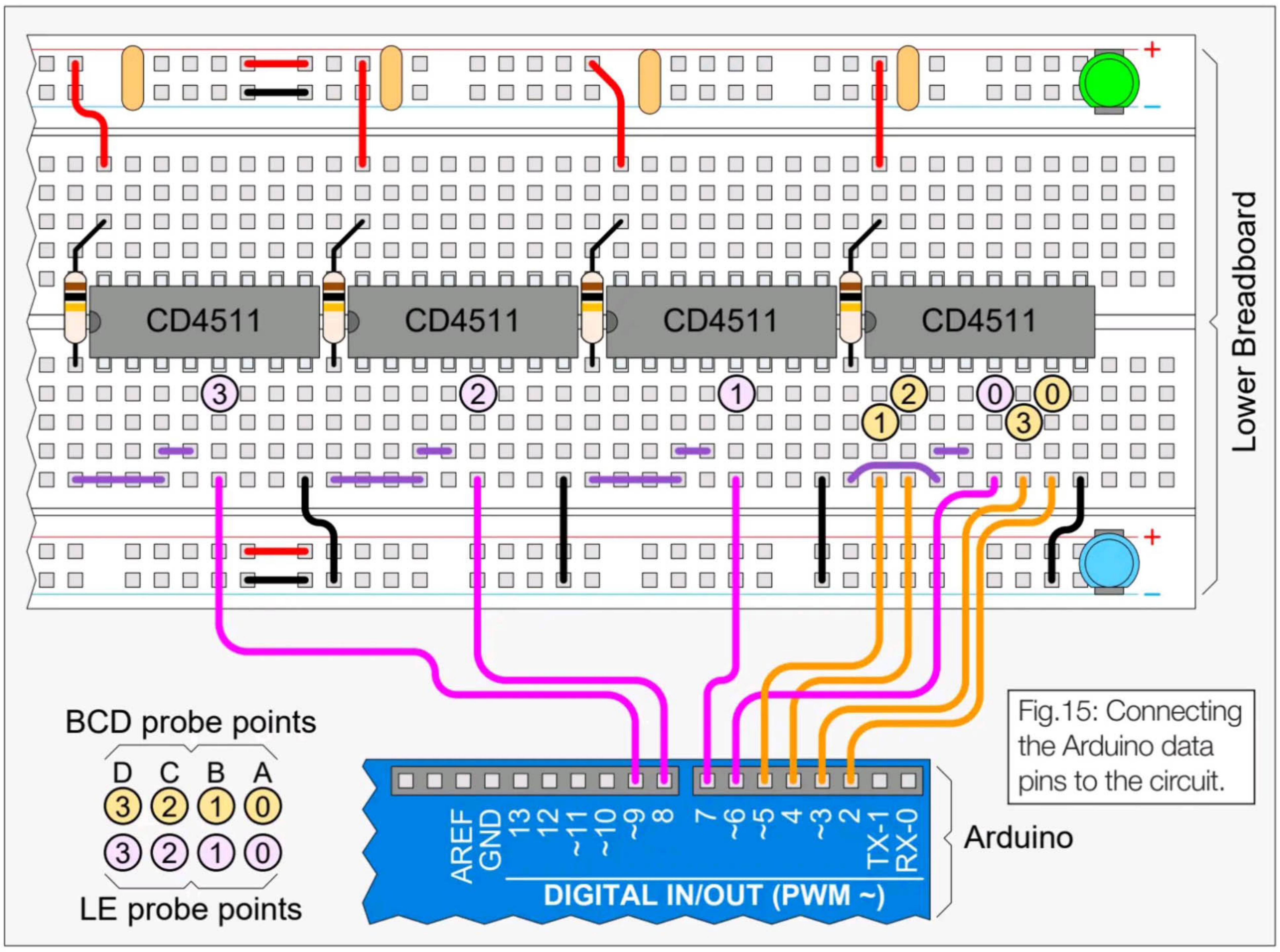

+
–
CD4511
CD4511
CD4511
CD4511
3
2
1
2
1
0
0
3
Lower Breadboard
BCD probe points
D C B A
3 2 1 0
3 2 1 0
LE probe points
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX-1
RX-0
DIGITAL IN/OUT (PWM ~)
Arduino

Fig.15: Connecting the Arduino data pins to the circuit.

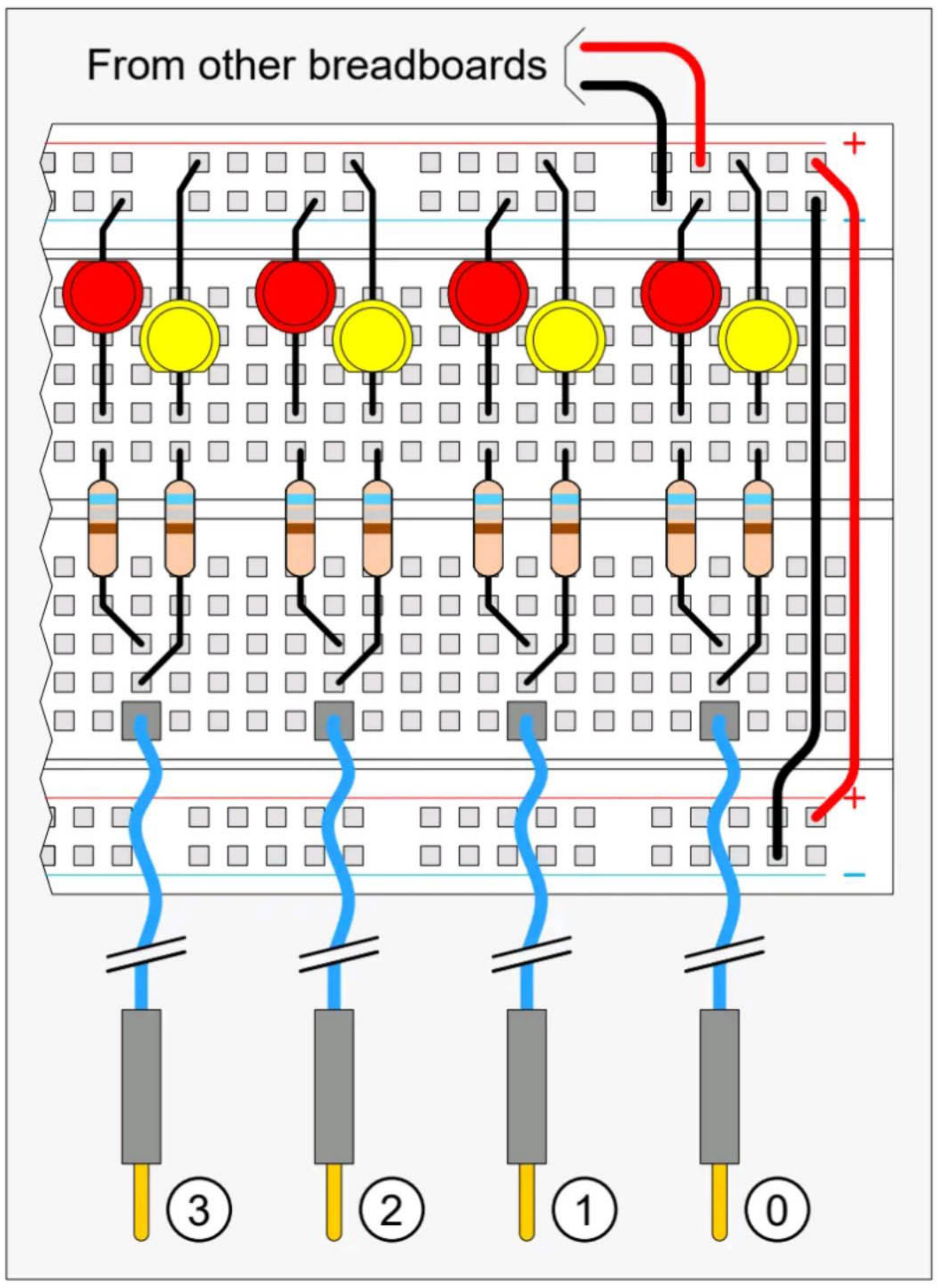
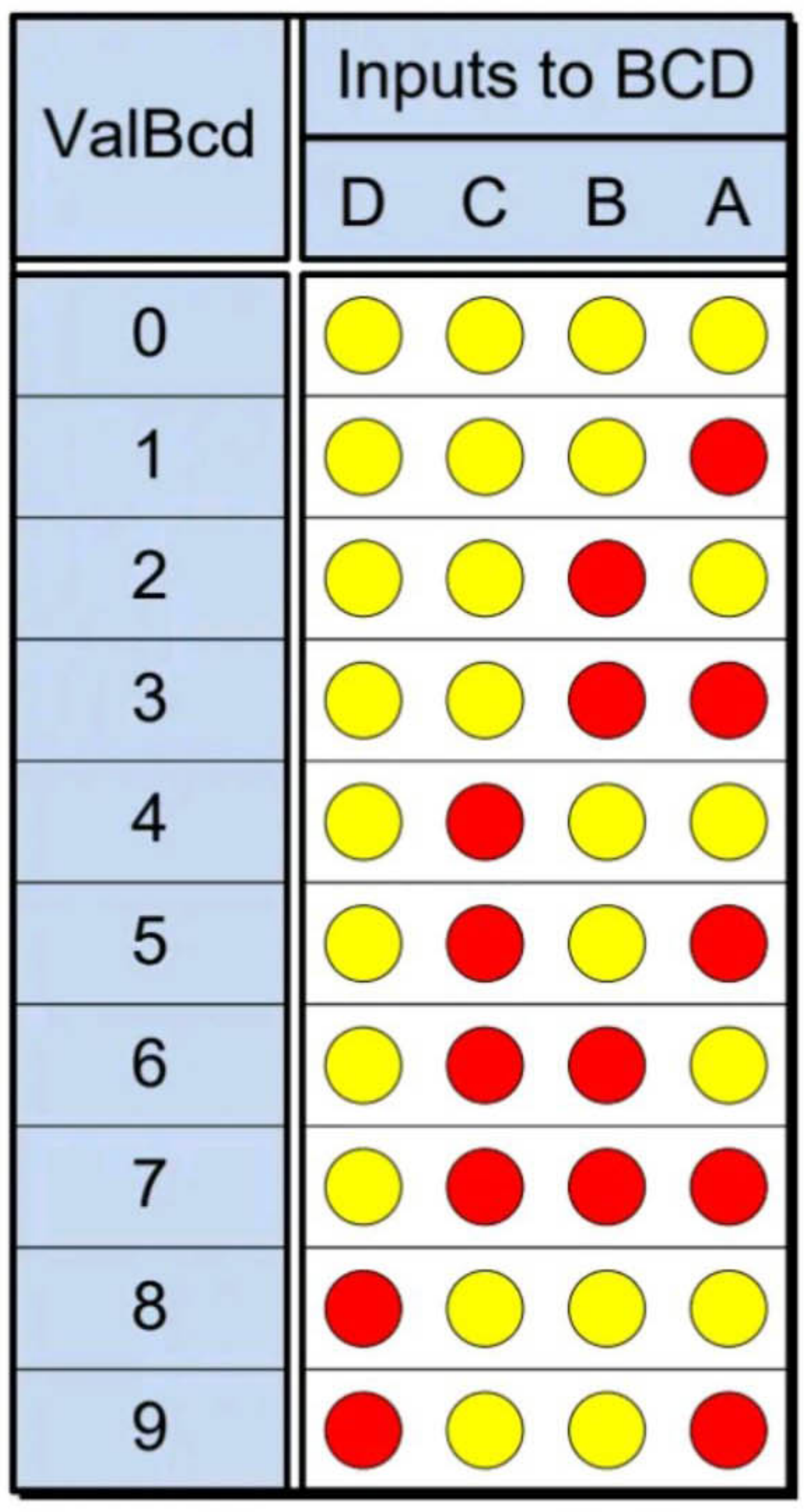
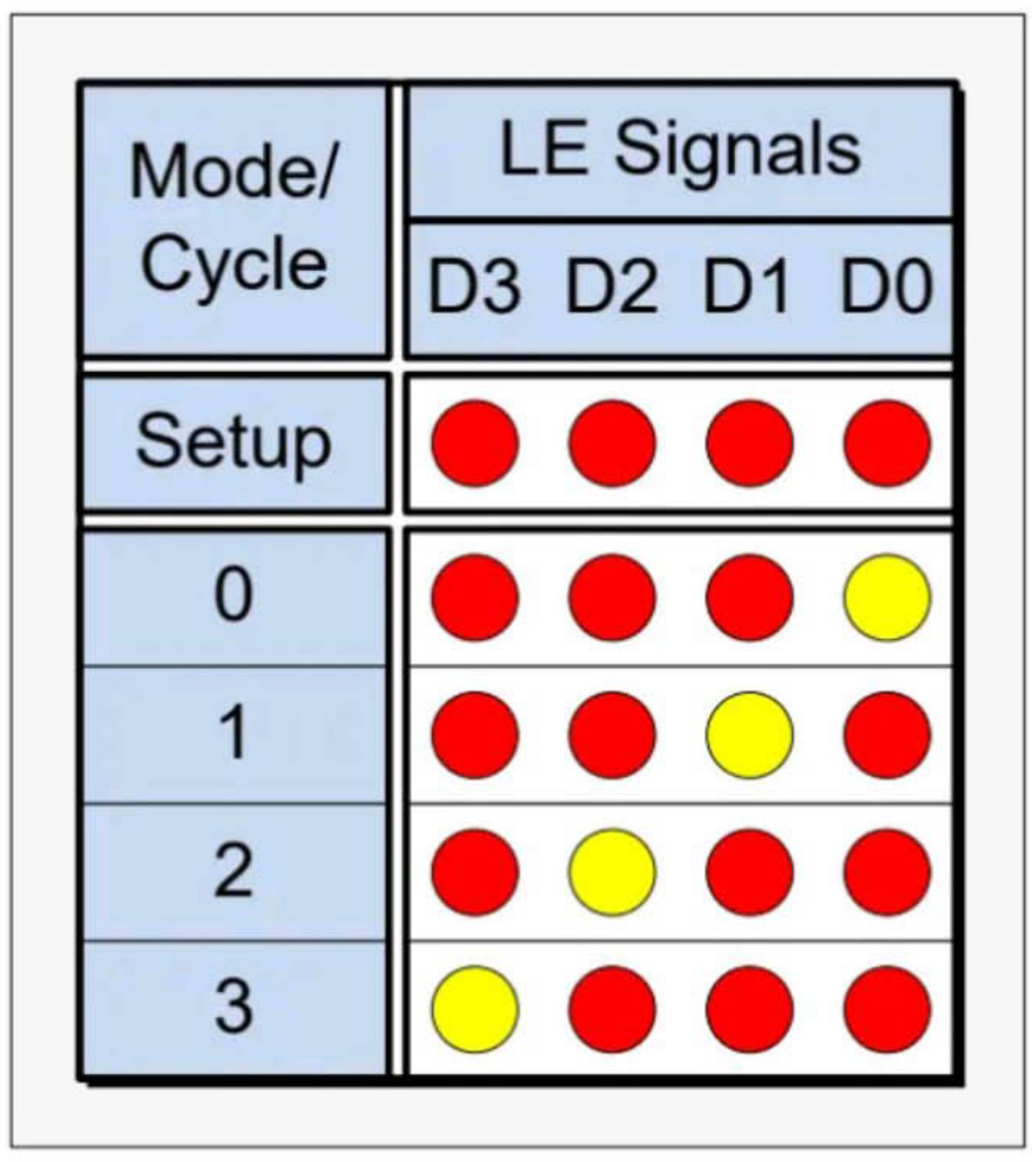


Fig.16: Making our four basic logic probes.



ValBcd	Inputs to BCD			
	D	C	B	A
0	🟡	🟡	🟡	🟡
1	🟡	🟡	🟡	🔴
2	🟡	🟡	🔴	🟡
3	🟡	🟡	🔴	🔴
4	🟡	🔴	🟡	🟡
5	🟡	🔴	🟡	🔴
6	🟡	🔴	🔴	🟡
7	🟡	🔴	🔴	🔴
8	🔴	🟡	🟡	🟡
9	🔴	🟡	🟡	🔴

Fig.17: The expected BCD data values as shown on the logic probes when running the test program. If you get this sequence, everything is working correctly so far!



Mode/ Cycle	LE Signals			
	D3	D2	D1	D0
Setup	🔴	🔴	🔴	🔴
0	🔴	🔴	🔴	🟡
1	🔴	🔴	🟡	🔴
2	🔴	🟡	🔴	🔴
3	🟡	🔴	🔴	🔴

Fig.18: The BCD LE signals.

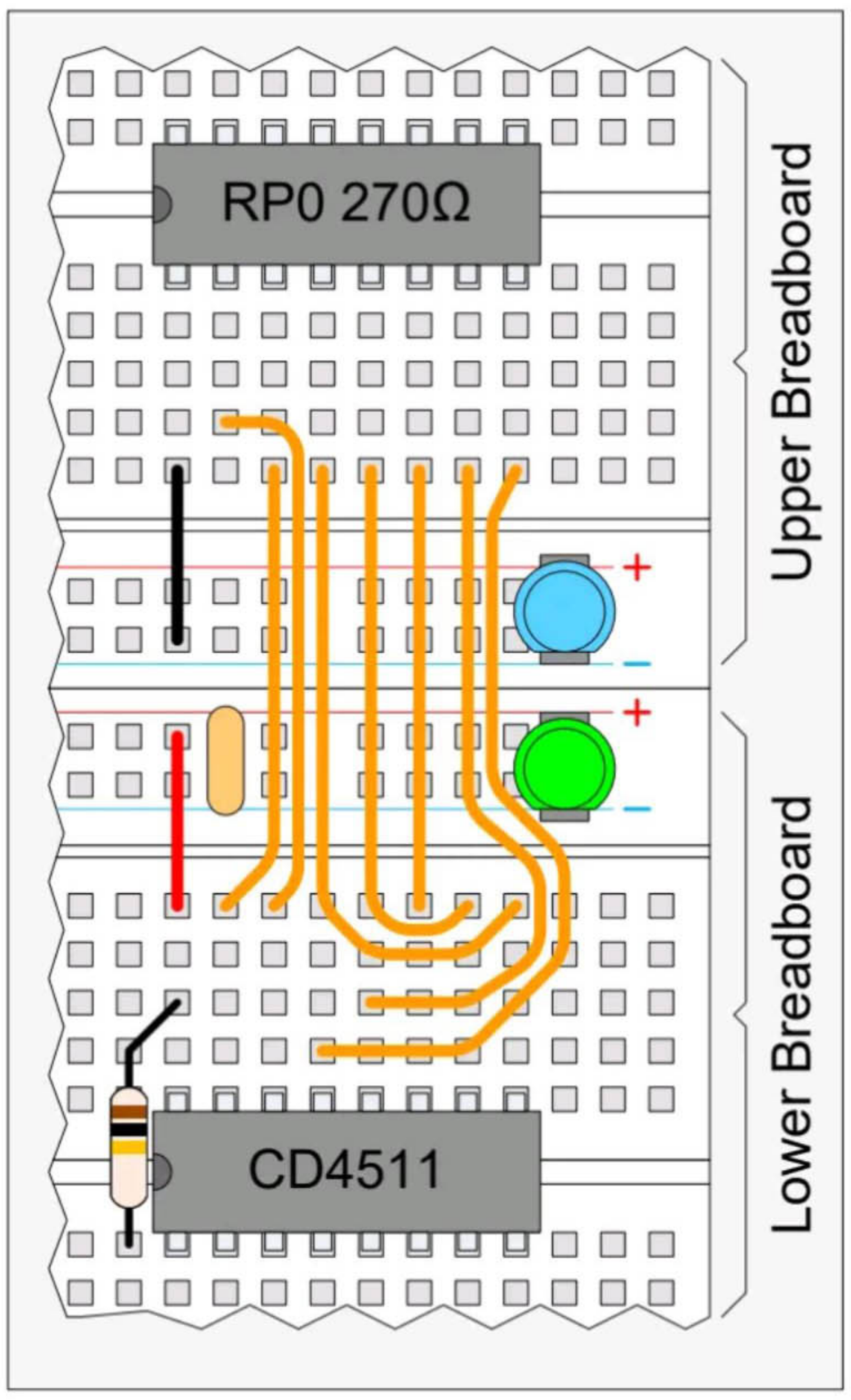


Fig.19: Connect the decoder to the resistors.

Stop! Before you power everything up, double-check that you've connected 0V to 0V and 5V to 5V. You'd be surprised how easy it is to slip up here. In the best case, reversing the supply polarity will mean it won't work, but it sometimes is possible to do some damage this way. So it's best to make a habit of double-checking this sort of thing before you proceed.

Do you remember the simple program we created in our previous column (*PE*, October 2024)? I'm talking about the one than repeatedly counts from 0 to 9, pausing for a second between counts while it presents the current count value on the Arduino's BCD outputs.

Well, let's take this program and tweak it a little to accommodate things like setting the CD4511's `LE` input to logic 1. You can find a copy of this updated program in the file named **CB-nov24-code-01.txt**.

When you power up your Arduino and load and run this program, you should see the BCD count sequence being displayed on your logic probe board as illustrated in Fig.17.

Now let's use this program as a starting point to create a new version that loops around applying a negative-going (1→0→1 ⊔) pulse to each of the CD4511's `LE` inputs in turn. A copy of this new code is available in the file named **CB-nov24-code-02.txt**.

Power everything down, remove probes 0, 1, 2, and 3 in Fig.16 from the BCD probe points shown in Fig.15, then re-connect them to the `LE` probe points shown in Fig.15. When you power everything up and load and run the new program, you should be presented with the sequence illustrated in Fig.18 on your logic probe board.

Finishing wiring up display 0

We are so close to finishing that I can taste it. Power everything down and connect the right-most CD4511 to resistor pack RP0 as illustrated in Fig.19. A full version of the latest breadboard setup is available to download in the file named **CB-nov24-brd-08.pdf**.

Now, let's create a new version of our program that loops around generating a random number between 0 and 9, presenting this number to the BCD inputs to our CD4511, pulsing the `LE` input low to load this value into the chip, and pausing for a second before doing it all again.

We'll also include a test to ensure that the new random number isn't the same as the old one. A copy of this new program is available in the file named **CB-nov24-code-03.txt**.

When you look at this program, you'll see that, even though we currently have only one CD4511 decoder and one 7-segment display fully wired up, the program attempts to load random values into all four decoders.

The reason for this is so that we don't have to modify the program when we get around to connecting our remaining displays.

Power everything up, load and run our new program and observe the random values appearing on display D0 (I could watch this running for hours!).

Connecting displays 1, 2, and 3

It's time to add the remaining wires. Power everything down, then connect resistor packs RP1, RP2, and RP3 to displays D1, D2, and D3, respectively. Next, connect the resistor packs to their corresponding CD4511s.

It's easy to get excited at this point – I know I did – resulting in me forgetting to add the wires connecting the `ABCD` inputs on the right-most CD4511 to the `ABCD` inputs on the remaining CD4511s.

I felt like a silly sausage when things failed to work as planned. It took me a while to realize what I'd done wrong (or neglected to do right), but I'm sure this won't happen to you.

If, unlike me, you manage to get the wiring right the first time, you will end up with a configuration like

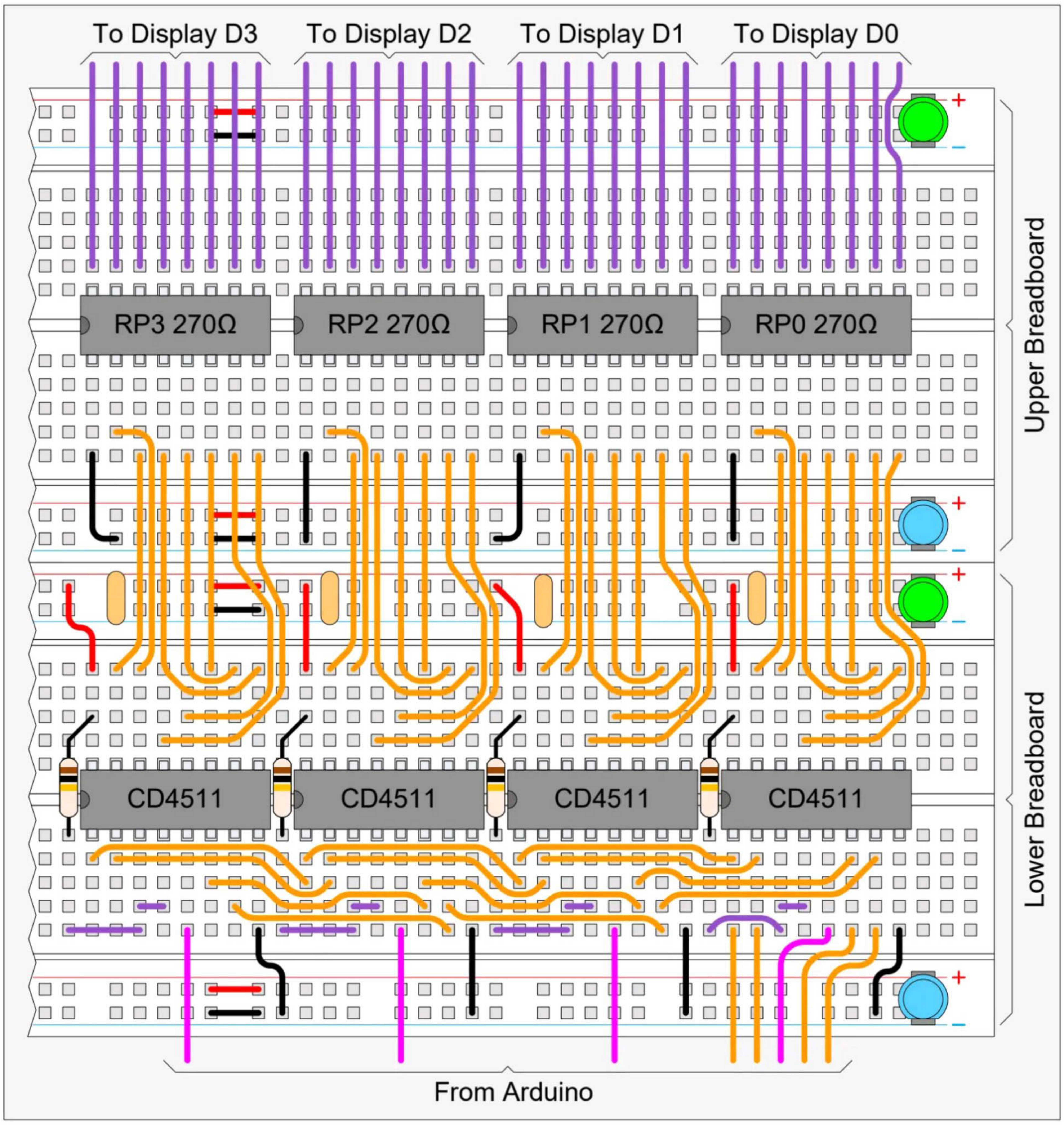

To Display D3
To Display D2
To Display D1
To Display D0
+
−
RP3 270Ω
RP2 270Ω
RP1 270Ω
RP0 270Ω
Upper Breadboard
CD4511
CD4511
CD4511
CD4511
Lower Breadboard
From Arduino

Fig.20: Connecting the remaining displays to our four CD4511 BCD decoders.

that shown in Fig.20 (a full drawing can be found in the file named **CB-nov24-brd-09.pdf**).

Remember that our current program is still loaded in the Arduino. This means that when we power everything up again, we should see different random values appearing on each of our displays.

Connecting the RTC

This is it, the final step! Power everything down and add the DS3231 RTC BOB to the left-hand side of the lower breadboard as illustrated in Fig.21. A full drawing of our final setup can be found in the file named **CB-nov24-brd-10.pdf**.

Remember to add the power and ground wires as shown. Also, remember to add the I^{2}C communications bus wires between the RTC and the Arduino as shown (SCL = clock, SDA = data).

As we discussed deep in the mists of time, the two Arduino pins we're using are connected in the board to its A4 and A5 analog inputs. That means we can't use the A4 and A5 pins for anything else if we're using them for the I^{2}C protocol.

All that remains is to modify our old 7448-based clock program to reflect our new CD4511-based implementation. I just did so.

You can download and peruse my version of this code at your leisure, in the file named **CB-nov24-code-04.txt**. All I can say is I'm watching my clock perform its magic and I'm squealing with delight (well, I'm squealing inside my head because I don't want to worry my wife).

This should keep you busy until next time when… but no… let's not let any spoilers ruin our excitement!

As always, I welcome your insightful comments, penetrating questions and any suggestions. **PE**

Components for Part 19	
Prototyping boards	https://bit.ly/3UMkcZ1
Kit of popular SN74LS00 chips	https://bit.ly/3wqgzyv
Components for Part 20	
16V 100µF electrolytic capacitors	https://bit.ly/44LzpNa
Ceramic capacitors (assorted values)	https://bit.ly/4bEAUiv
Components for Part 22	
SN74LS48N BCD Decoders	https://bit.ly/3zT18jx
16-pin resistor pack (8 × 390Ω)	https://bit.ly/4d0ISDz
Components for Part 24	
4 × CD4511 BCD Decoders	https://bit.ly/3yzWl6k
16-pin resistor pack (8 × 390Ω)	https://bit.ly/46HZ2PQ

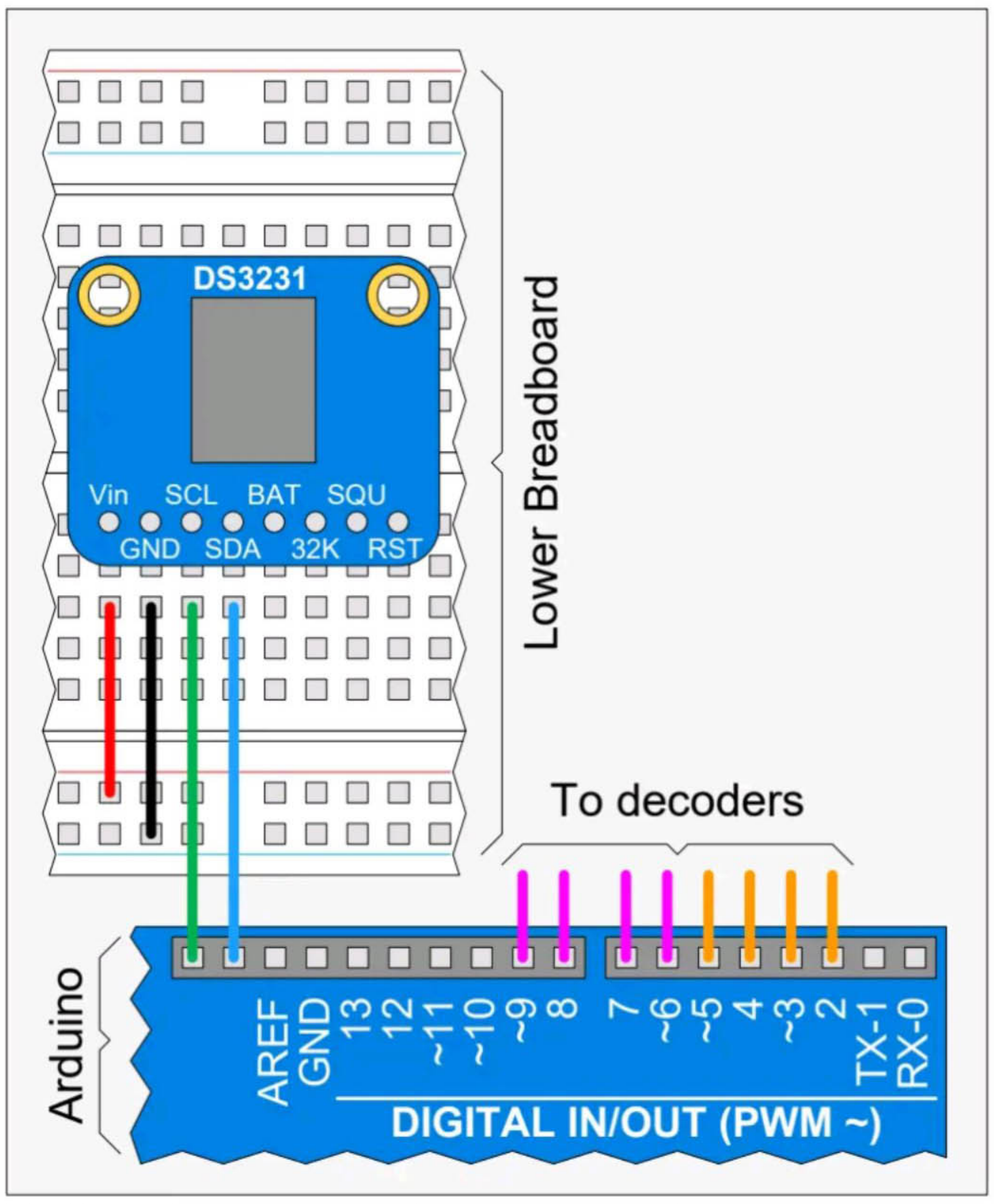
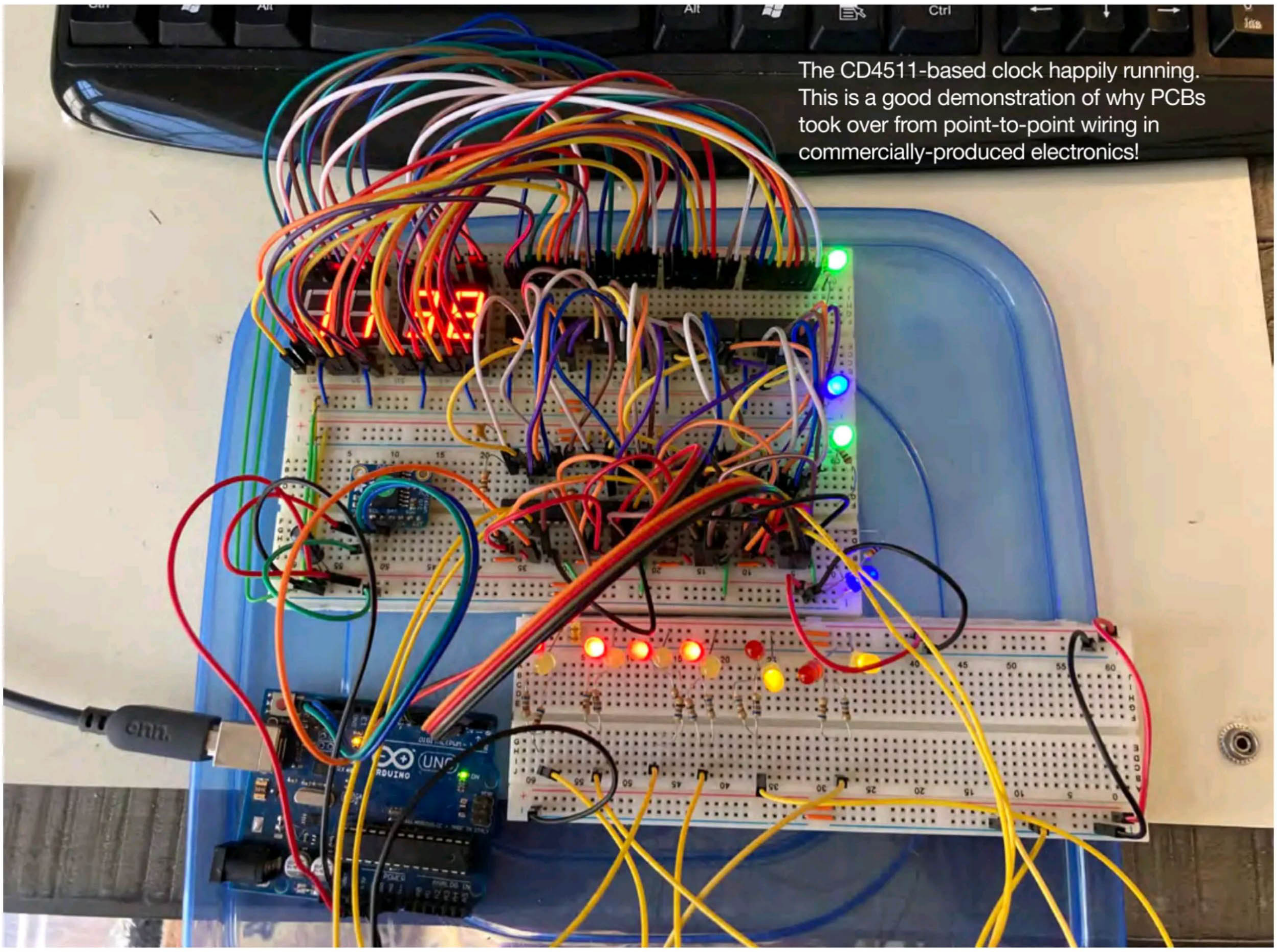

DS3231
Vin
SCL
BAT
SQU
GND
SDA
32K
RST
Lower Breadboard
To decoders
Arduino
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX-1
RX-0
DIGITAL IN/OUT (PWM ~)

Fig.21: Adding the real-time clock (RTC).



The CD4511-based clock happily running. This is a good demonstration of why PCBs took over from point-to-point wiring in commercially-produced electronics!

Teach-In 8 CD-ROM Exploring the Arduino

This CD-ROM version of the exciting and popular *Teach-In 8* series has been designed for electronics enthusiasts who want to get to grips with the inexpensive, immensely popular Arduino microcontroller, as well as coding enthusiasts who want to explore hardware and interfacing. *Teach-In 8* provides a one-stop source of ideas and practical information.

The Arduino offers a remarkably effective platform for developing a huge variety of projects; from operating a set of Christmas tree lights to remotely controlling a robotic vehicle wirelessly or via the Internet. *Teach-In 8* is based around a series of practical projects with plenty of information for customisation. The projects can be combined together in many different ways in order to build more complex systems that can be used to solve a wide variety of home automation and environmental monitoring problems. The series includes topics such as RF technology, wireless networking and remote web access.

PRICE £8.99

Includes P&P to UK if ordered direct from us

PLUS: PICs and the PICkit 3 – A beginners guide

The CD-ROM also includes a bonus – an extra 12-part series based around the popular PIC microcontroller, explaining how to build PIC-based systems.

SOFTWARE

The CD-ROM contains the software for both the *Teach-In 8* and PICkit 3 series.

ORDER YOUR COPY TODAY!
JUST CALL 01202 880299 OR VISIT www.electronpublishing.com

Part 1 by Tim Blythman

Multi-Channel Volume Control

Many people now have home theatre/surround sound systems and need to control the volume of six or more audio channels. This Multi-Channel Volume Control has a touchscreen and receives infrared remote signals. It can be expanded up to 20 channels, although six or eight will suit most applications.

SILICON CHIP
VOLUME
MULTI-CHANNEL CONTROL
1.5dB
EEPROM
MUTE
DOWN
UP
MUTE
SETUP

The Multi-Channel Volume Control can be built as a stand-alone unit in its own enclosure, or as a modular system that can be incorporated into a multi-channel preamplifier or amplifier. It can control up to 20 channels in synchrony.

A microcontroller senses inputs from the touchscreen, rotary encoder or infrared remote control and drives digital potentiometers to control the volume of each channel.

One of the reasons behind this project was our publication of the Hummingbird Amplifier module in the December 2022 issue. That small, low-cost 100W amplifier module makes it easy to build an amplifier with four, six or even more channels. You need something like this design to adjust the volume of all those modules together.

The alternative would be a four-ganged or six-ganged potentiometer, but then you would have poor tracking and messy wiring. Pots can also go scratchy after a while, unlike a digital pot, which generally continues working flawlessly for decades.

This design is based on our previous Touchscreen Digital Preamp (September & October 2022 issues), which only sports two channels and is not expandable.

Besides this design, another option is Phil Prosser's "The Digital Potentiometer" (March 2024). While it is a two-channel design, it does allow you to gang up multiple boards. However, it is a more 'bare bones' design than this one, lacking the touchscreen option or any onboard audio sockets.

Performance

The Multi-Channel Volume Control is based on a Baxandall-style volume control circuit but uses a digital potentiometer rather than the regular

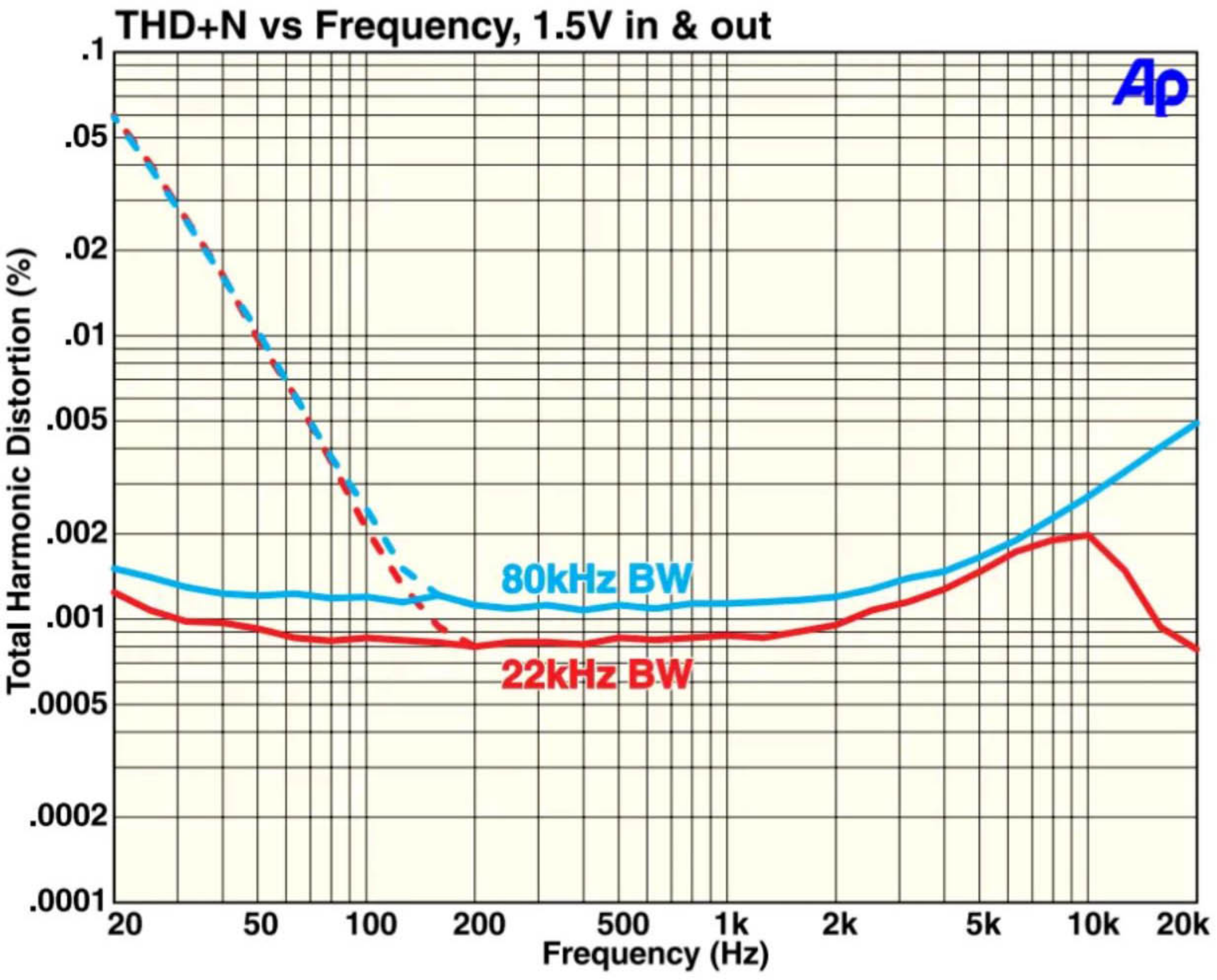


Fig.1: the performance is similar to the Touchscreen Digital Preamp, with THD+N typically less than 0.002% across much of the range. The dashed lines show the degradation with 1μF ceramic coupling caps instead of 2.2μF tantalums.

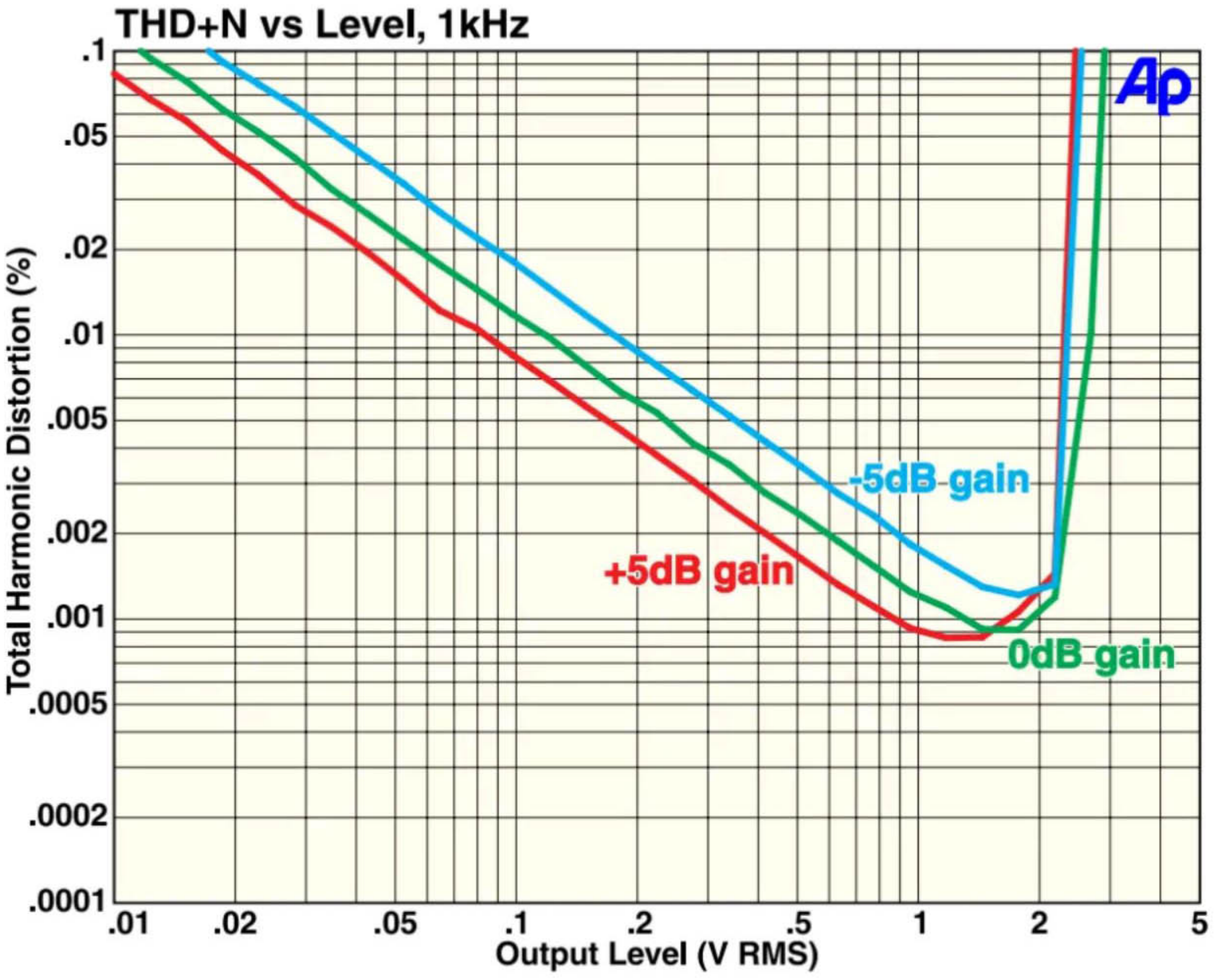


Fig.2: a plot of THD+N against input level. The sweet spot for input levels is ~1.5V, but anywhere in the range of 1-2.3V RMS is fine. Above 2.3V RMS starts to cause clipping while lower levels suffer due to the closer noise floor.

Features & Specifications

- Can control the volume of four, eight, 12, 16 or 20 channels
- RCA sockets for inputs and outputs
- Volume levels/gain settings: mute (-100dB) and -48dB to +16dB
- 1dB or smaller steps from -30dB to +16dB
- Digital controls, including touchscreen, rotary encoder and IR remote
- 2.8in LCD screen shows the volume and mute status
- Can be programmed to support many NEC-compatible IR codes
- Each channel can have a preset offset applied
- Volume and mute settings kept in EEPROM for next power on
- Optional small OLED status display with rotary encoder
- THD+N: typically less than 0.002% (see Fig.1)
- Channel separation: >80dB (see Fig.3)
- Signal handling: 0.1-2.5V RMS

Image source: unsplash.com/photos/HLhmbBw6xpY

kind. Handily, this circuit provides a logarithmic response from a linearly changing resistance, making it easy to map the volume settings to levels in decibels.

Figs.1-4 show the resulting performance. Fig.1 plots total harmonic distortion plus noise (THD+N) against frequency for an input signal level of 1.6V RMS. As you'd expect, the plots are similar to those for the Touchscreen Digital Preamp.

The red plot was taken with a 20Hz-22kHz bandwidth, which best represents the normal audible range and thus what you would hear. The cyan/blue plot was taken with a wider bandwidth, up to 80kHz, which includes the harmonics of higher-frequency signals. They are not directly audible but could intermodulate to affect audible frequencies.

As you can see, THD+N is below 0.001% for up to 2kHz and below 0.002% up to about 7kHz.

Fig.2 shows the total harmonic distortion plus noise (THD+N) against signal level for three volume (gain) settings. The distortion is higher at lower signal levels as the fixed noise dominates the smaller signals. The sharp rise around 2.5V RMS is where the Volume Control enters clipping.

The sweet spot is with an input signal around 1.5V RMS.

Fig.3 shows the crosstalk between channels. The two plots show the extremes that can be expected within a single Volume Module (explained later). The cyan/blue plot shows the crosstalk from adjacent channels and is around -80dB or better; these channels share some components. The red plot shows the separation between channels at opposite ends of a module, which is a little better.

Fig.4 shows the frequency responses at three gain settings: +5dB (red), 0dB (green) and -5dB (cyan/blue). There is a slight but uniform roll-off at lower frequencies, but the level difference is uniform across the audible band. In other words, the volume adjustment is consistent, as is expected.

Modular design

While we are boasting up to 20 channels, most constructors will not need that many. So rather than offering a design based around a single PCB, the Volume Control is modular. The three modules are the Control/Power Supply Module, the four-channel Volume Module and an optional OLED Module.

The main Control and Power Supply Module incorporates a microcontroller that converts the user input into the necessary actions to implement the volume control. It also contains all the power supply circuitry needed to drive any other connected modules.

The power supply section only needs a 12V AC supply, which can be provided from a small mains transformer or AC plugpack if your system doesn't already have a suitable source. You probably won't need a separate transformer if your system has a larger transformer incorporating 12V AC taps.

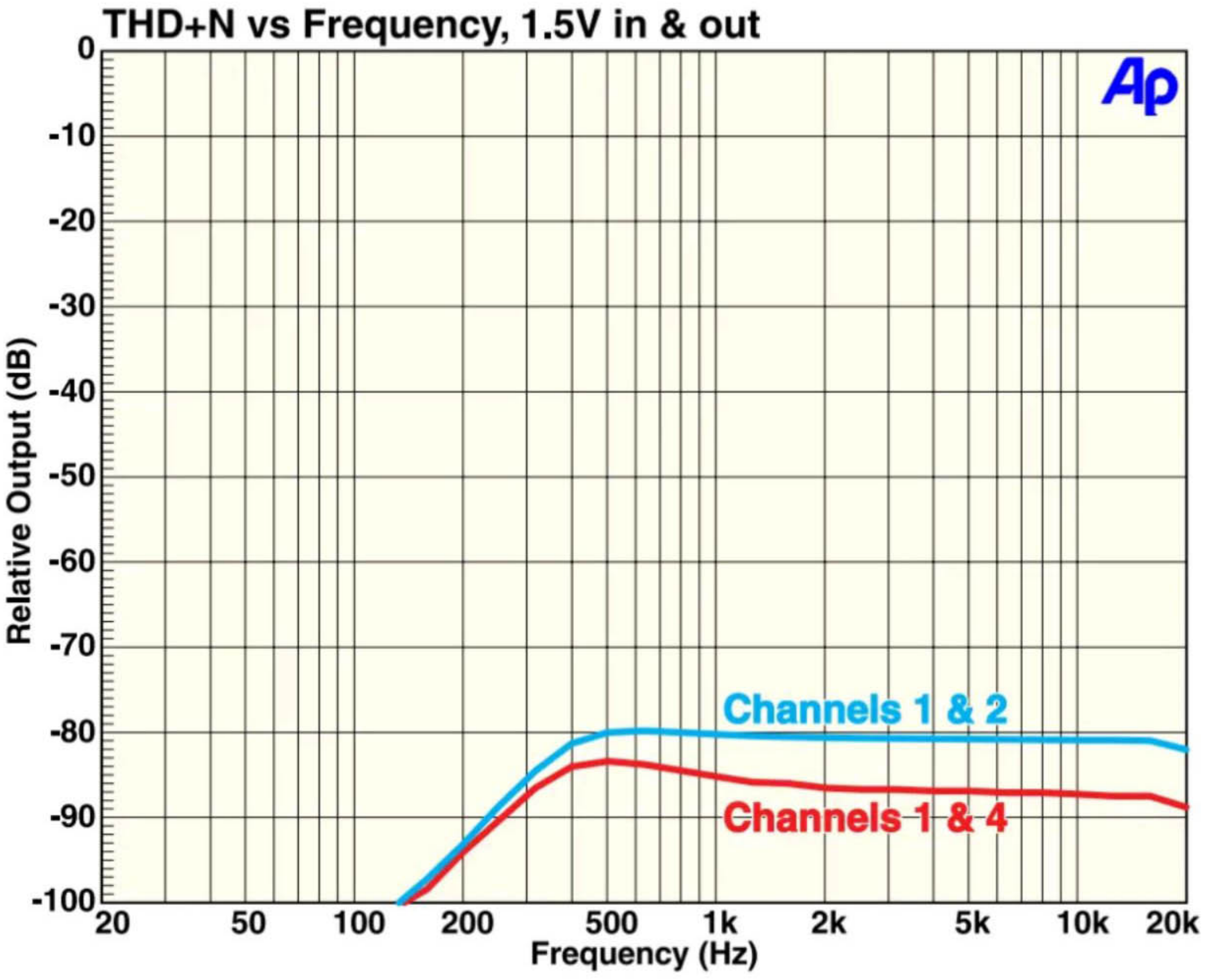


Fig.3: channel separation is about 80dB at worst and only for adjacent channels that share an op amp in their audio paths.

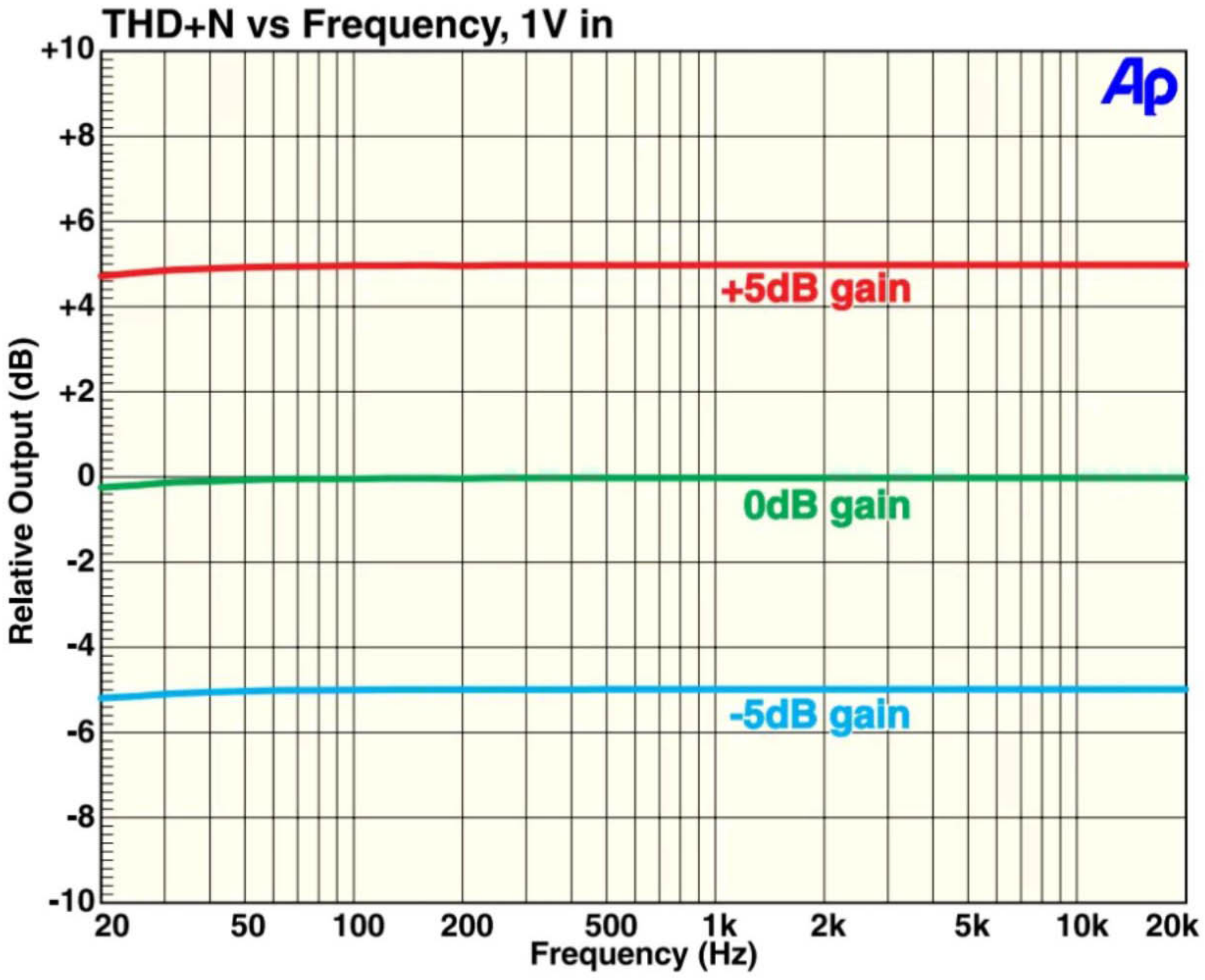
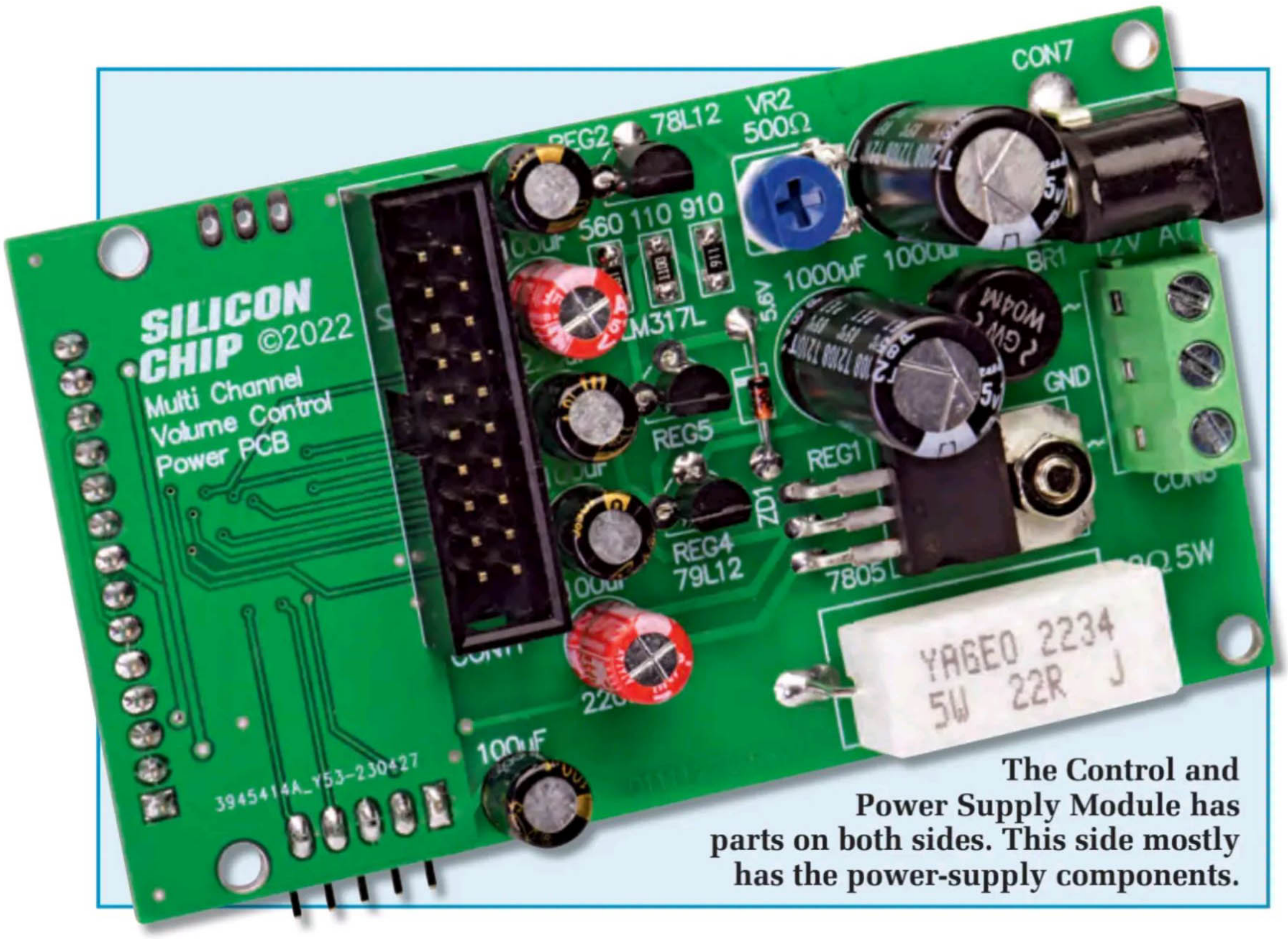


Fig.4: a frequency response plot at three different volume settings (-5dB, 0dB and +5dB). There is only a slight roll-off at the bottom end, and the effect of the volume settings is very uniform across the spectrum.



The Control and Power Supply Module has parts on both sides. This side mostly has the power-supply components.

Although we haven't tested it, a pair of ±15V DC rails could be fed to the Control and Power Supply Module instead.

The Control/Power Supply module also incorporates an infrared receiver and a 2.8-inch LCD touchscreen. The Module's PCB is the same size as the LCD panel at 50mm tall, making it a comfortable fit for a 3U rack case, or anything taller.

The Volume Module can control the volume of up to four channels and incorporates an AD8403 precision quad digital potentiometer. This Module also has four input and four output RCA sockets, plus other components to buffer and drive the audio signal as it passes through.

A third type of module, the OLED Module, is a simple and compact alternative (or supplement) to the touchscreen display. It includes a small OLED screen for those who want something smaller and simpler; its input control is a rotary encoder.

The various modules are connected by a ribbon cable punctuated by IDC connectors. This carries all power and control signals between the modules.

The simplest configuration is a single Control and Power Supply Module with between one and five Volume Modules, allowing 4-20 channels to have their levels adjusted as they pass through.

If the OLED Module is added, only four volume modules can be connected, which limits the number of controlled channels to sixteen. Still, we don't see that as a major limitation!

Due to the various ways this project can be arranged, and the fact that many will be building it as part of a larger system, we will not specify a particular enclosure. You can choose an enclosure based on your requirements that will also fit any amplifier modules, transformers and other necessary bits.

We'll start by describing how each module works. Then, next month, we'll follow up by outlining the assembly of each type of module, along with instructions on how they are wired together, tested and used.

Control and Power Supply Module

Fig.5 is the circuit diagram for the Control and Power Supply Module. It receives 12V AC via CON7 or CON8.

CON7 is a barrel socket and thus can only accept a single 12V AC input. On the other hand, CON8 has three terminals and is intended to connect to a 24V AC centre-tapped transformer, or two 12V AC phases; however, you could connect a single winding via CON8 if that's all you had.

Either way, the AC supply passes through bridge rectifier BR1 and a pair of 1000µF electrolytic capacitors to provide a nominally ±17V DC supply to be regulated. Feeding in two AC phases via CON8 is preferred as the capacitors only need to hold up through the 10ms of each half-cycle rather than 20ms for a full mains cycle.

The board generates five regulated DC rails. REG2 (78L12) and REG4 (79L12) provide +12V and -12V, respectively, to power the analog circuitry. Their outputs are filtered by 100µF capacitors and taken to CON11, which feeds the ribbon cable bus noted earlier, for distribution to the other modules.

The +12V rail is further regulated to 5.5V by REG5, an LM317L and its accompanying components. The accompanying resistors and VR2 allow its output voltage to be trimmed to account for resistor tolerances.

A 220µF capacitor sits on the output of REG5, and this voltage is also taken to CON11. If the 5.5V rail is adjusted too high, zener diode ZD1 conducts and protects other circuitry downstream.

The unregulated positive rail is also dropped via a 22Ω 5W resistor (to spread dissipation) and reduced to 5V by REG1 (7805). It has a 100µF bypass capacitor on its input and a 220µF filter capacitor on its output.

This 5V rail is used on the Control and Power Supply Module to power the LCD touchscreen; the power needed by the LCD backlight, and the resulting higher dissipation, is the main reason why this regulator is a larger TO-220 type, while the others are in smaller TO-92 packages.

The 5V rail is reduced to 3.3V by REG3 (MCP1700) to power microcontroller IC9. The 3.3V rail is also available on CON11 to power the microcontroller on the OLED Module.

Broadly, the +12V and -12V rails power analog components, while the 5.5V rail primarily powers the digital potentiometers. The 5V and 3.3V rails power the digital components. The separate power domains help to minimise any intrusion of digital signals into the analog, which would affect audio quality.

On the Control and Power Supply Module, the remaining circuitry consists mainly of the microcontroller, the LCD touchscreen and their essential ancillaries. IC9 is a 20-pin, 8-bit PIC16F18146 microcontroller. It is powered from the 3.3V rail, so it can easily interface with the LCD controller, which also runs at 3.3V from its own regulator on the LCD's PCB.

A 10kΩ resistor pulls IC9's pin 4 $\overline{\text{MCLR}}$ pin to the 3.3V rail, preventing spurious resets, while a 100nF capacitor bypasses its supply to pins 1 and 20. These three pins, plus the

Multi-channel Volume Control, part 1

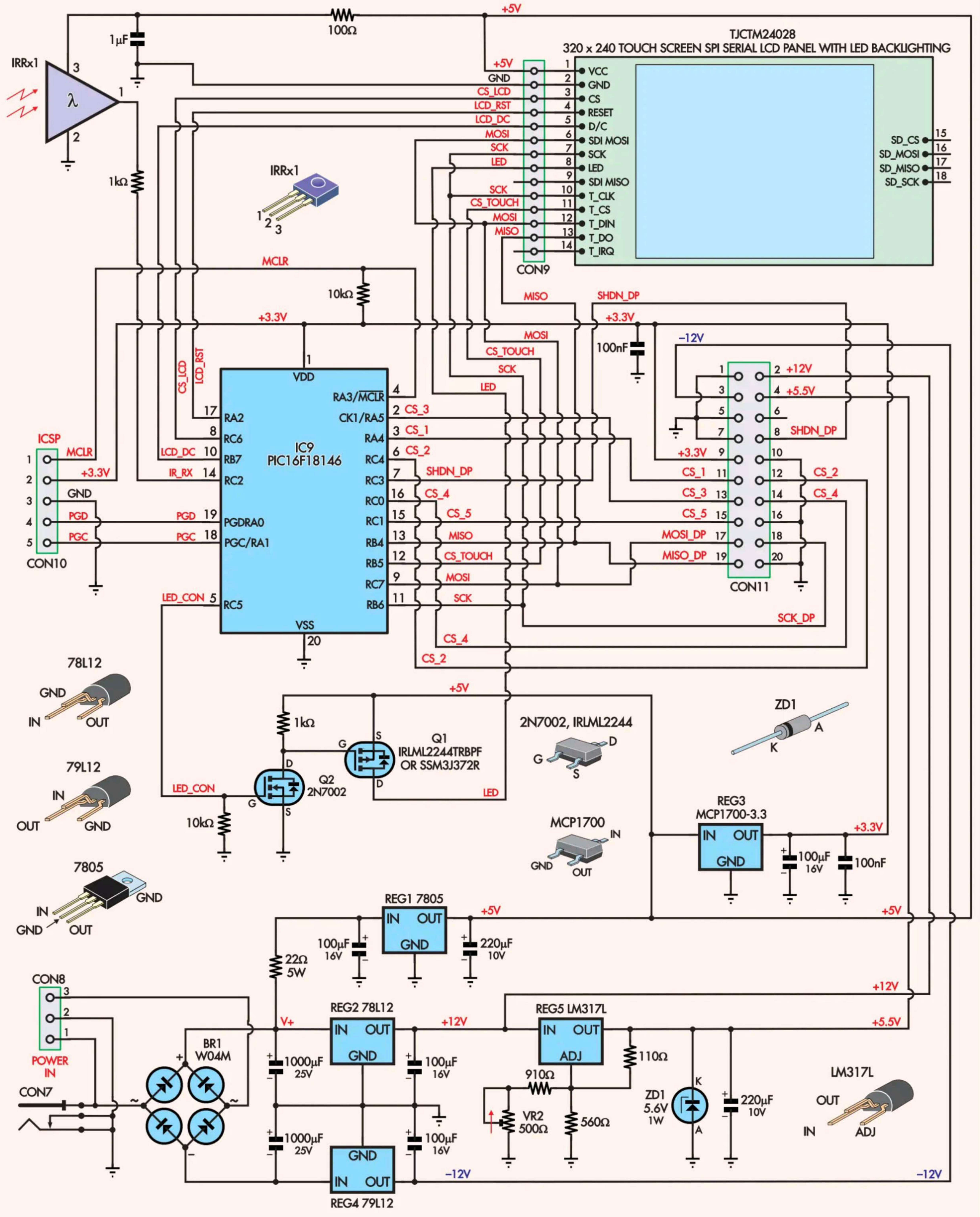

+5V
100Ω
1µF
IRRx1
λ
3
1
2
1kΩ
TJCTM24028
320 x 240 TOUCH SCREEN SPI SERIAL LCD PANEL WITH LED BACKLIGHTING
VCC
GND
CS
RESET
D/C
SDI MOSI
SCK
LED
SDI MISO
T_CLK
T_CS
T_DIN
T_DO
T_IRQ
CS_LCD
LCD_RST
LCD_DC
MOSI
SCK
LED
CS_TOUCH
MISO
CON9
SD_CS
SD_MOSI
SD_MISO
SD_SCK
MCLR
10kΩ
+3.3V
100nF
SHDN_DP
–12V
IC9
PIC16F18146
VDD
VSS
RA3/MCLR
CK1/RA5
RA4
RC4
RC3
RC0
RC1
RB4
RB5
RC7
RB6
RA2
RC6
RB7
RC2
PGDRA0
PGC/RA1
RC5
CS_1
CS_2
CS_3
CS_4
CS_5
IR_RX
PGD
PGC
LED_CON
ICSP
CON10
GND
+12V
+5.5V
MOSI_DP
MISO_DP
SCK_DP
CON11
78L12
79L12
7805
IN
OUT
GND
Q1
IRLML2244TBPF
OR SSM3J372R
Q2
2N7002
2N7002, IRLML2244
G
D
S
ZD1
A
K
MCP1700
REG3
MCP1700-3.3
100µF 16V
REG1 7805
220µF 10V
22Ω 5W
CON8
POWER IN
CON7
BR1
W04M
V+
REG2 78L12
REG5 LM317L
ADJ
1000µF 25V
110Ω
910Ω
VR2 500Ω
560Ω
ZD1 5.6V 1W
REG4 79L12
LM317L
SC ©2023
MULTI-CHANNEL DIGITAL VOLUME CONTROL
CONTROL & POWER SUPPLY MODULE

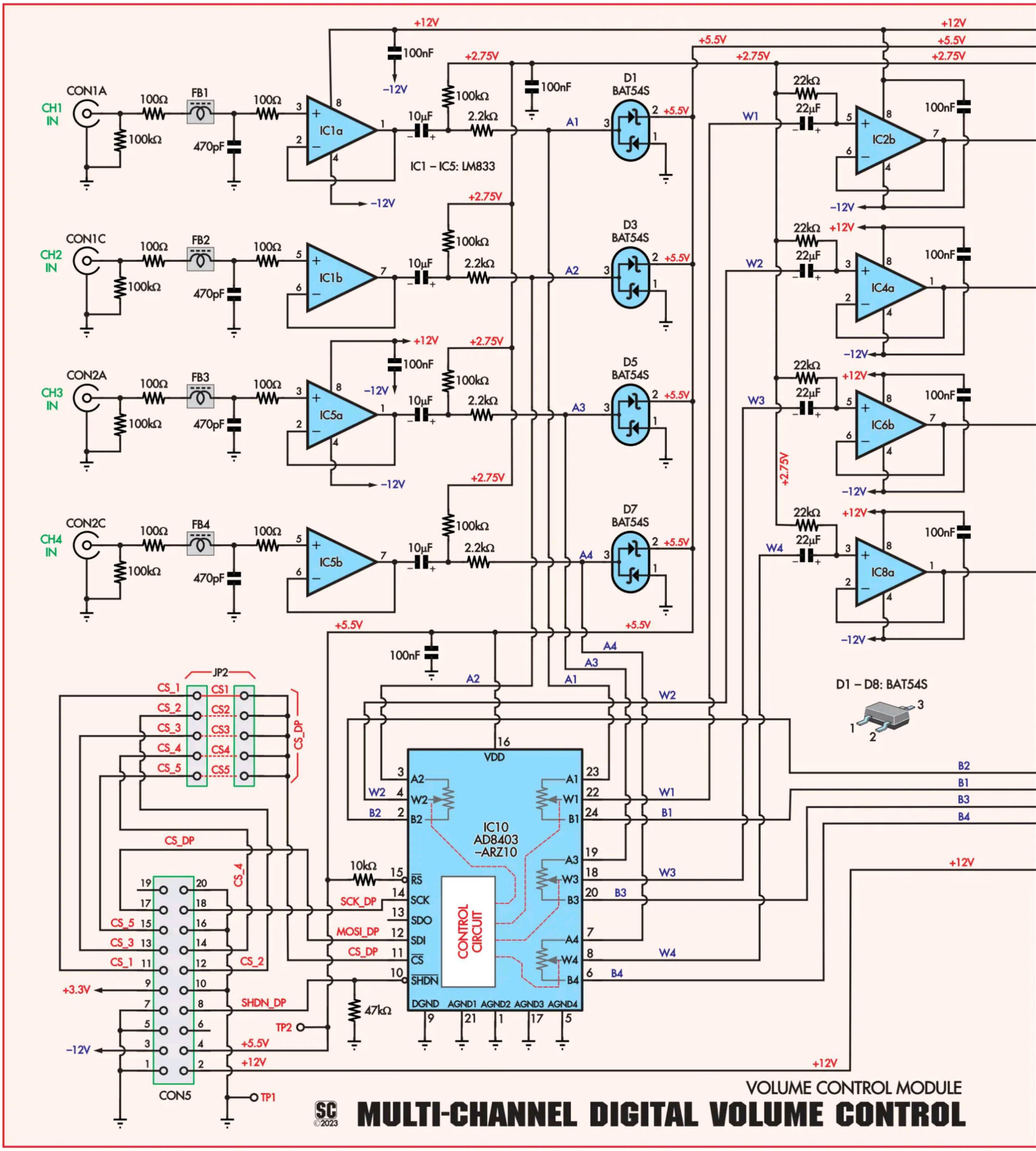
Fig.5: the Control and Power Supply Module circuit derives +12V, -12V, +5.5V, +5V and +3.3V rails to power the control circuitry and external modules. The microcontroller handles the LCD touch panel and receives inputs from an infrared receiver. It also sends signals to the two other module types over a 20-way bus via CON11.

PGC and PGD programming pins, go to ICSP header CON10 for in-circuit programming if needed.

Another nine of the micro's digital pins are wired to CON11 to control the other modules. The SCK, MOSI and MISO pins form the SPI serial bus that controls the other modules. The PIC16F18146 can remap its digital peripherals to any digital pin, so the pin allocation was chosen to simplify the PCB routing.

Five $\overline{\text{CS}}$ (chip select) lines are also broken out, allowing up to five different slave modules to be independently addressed for the SPI bus. The $\overline{\text{SHDN}}$ (shutdown) line allows the micro to signal to all Volume Modules that their outputs should be muted.

The SPI bus pins are also wired to the LCD touchscreen via CON9, along with the other four digital control lines it needs. A pair of Mosfets plus pullup and pulldown resistors power the LED backlight in the LCD touchscreen, controlled by a digital signal (LED_CON) from IC9.


CH1 IN
CON1A
CH2 IN
CON1C
CH3 IN
CON2A
CH4 IN
CON2C
100Ω
100kΩ
FB1
FB2
FB3
FB4
470pF
IC1a
IC1b
IC5a
IC5b
IC1 – IC5: LM833
+12V
–12V
+2.75V
+5.5V
100nF
10µF
2.2kΩ
D1 BAT54S
D3 BAT54S
D5 BAT54S
D7 BAT54S
A1
A2
A3
A4
W1
W2
W3
W4
B1
B2
B3
B4
22kΩ
22µF
IC2b
IC4a
IC6b
IC8a
D1 – D8: BAT54S
JP2
CS_1
CS_2
CS_3
CS_4
CS_5
CS1
CS2
CS3
CS4
CS5
CS_DP
CON5
+3.3V
SHDN_DP
TP1
TP2
47kΩ
10kΩ
SCK_DP
MOSI_DP
VDD
RS
SCK
SDO
SDI
CS
SHDN
IC10 AD8403 –ARZ10
CONTROL CIRCUIT
DGND
AGND1
AGND2
AGND3
AGND4
SC ©2023
VOLUME CONTROL MODULE
MULTI-CHANNEL DIGITAL VOLUME CONTROL

Infrared receiver IRRx1 takes its power from the 5V rail, smoothed by the 100Ω resistor and 1µF capacitor. Its output goes to the last of IC9's unused pins via a 1kΩ resistor.

These receivers typically have a 30kΩ internal pullup, so even though it is powered from 5V, it can safely interface to the microcontroller expecting 3.3V levels. That's because the 5V drive is quite weak and easily clamped by the microcontroller's internal input protection diodes.

Volume Module

The circuit diagram for a single Volume Module is in Fig.6. CON5 connects to the Control and Power Supply Module's CON11 via a 20-way ribbon cable.

An adjustable padded divider that includes trimpot VR1 is used to derive a 2.75V rail from the 5.5V supply. VR1 trims this voltage, which is bypassed by a 220µF capacitor. Test points are provided to allow easy measurement during setup. The 2.75V rail must be

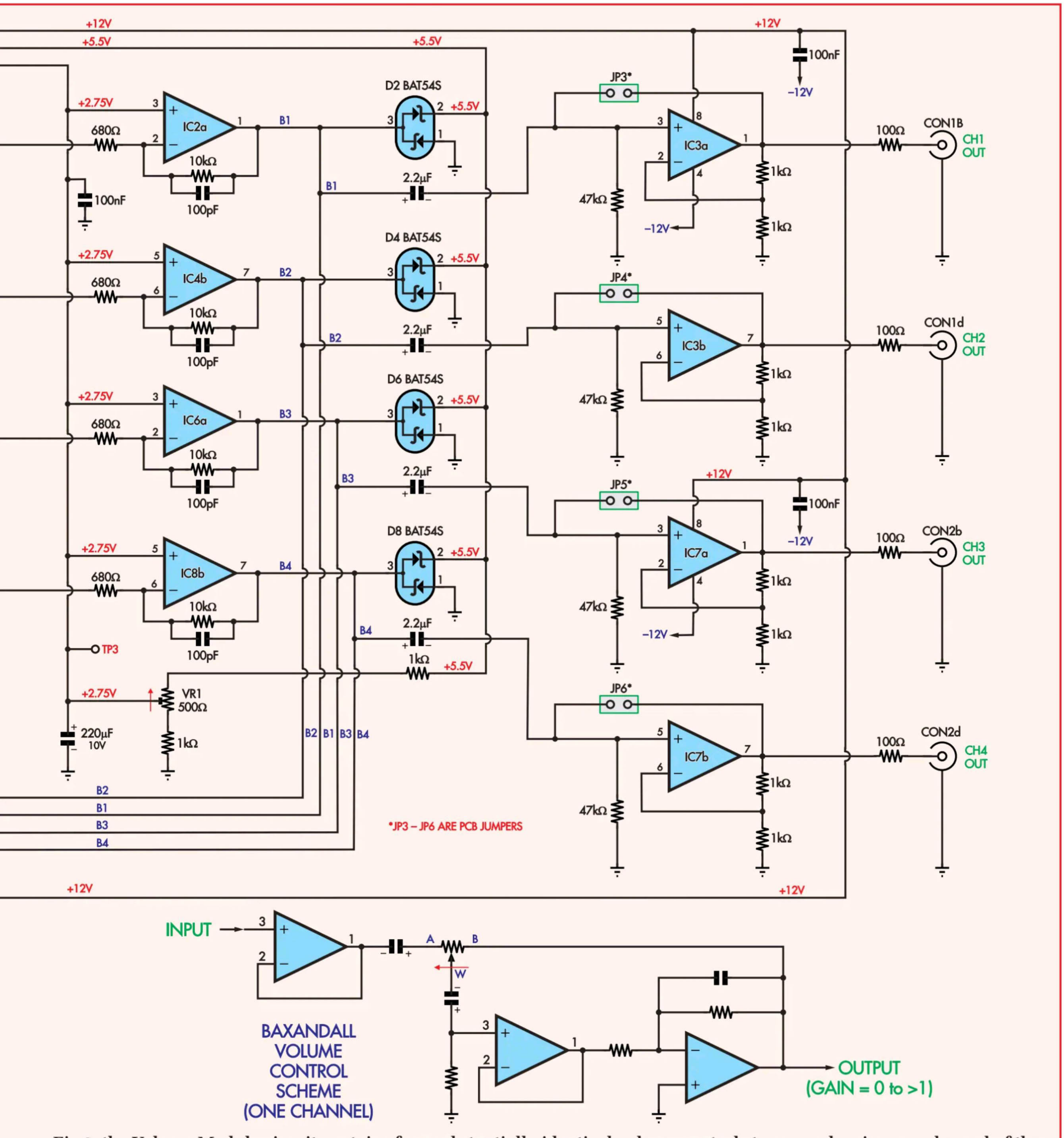
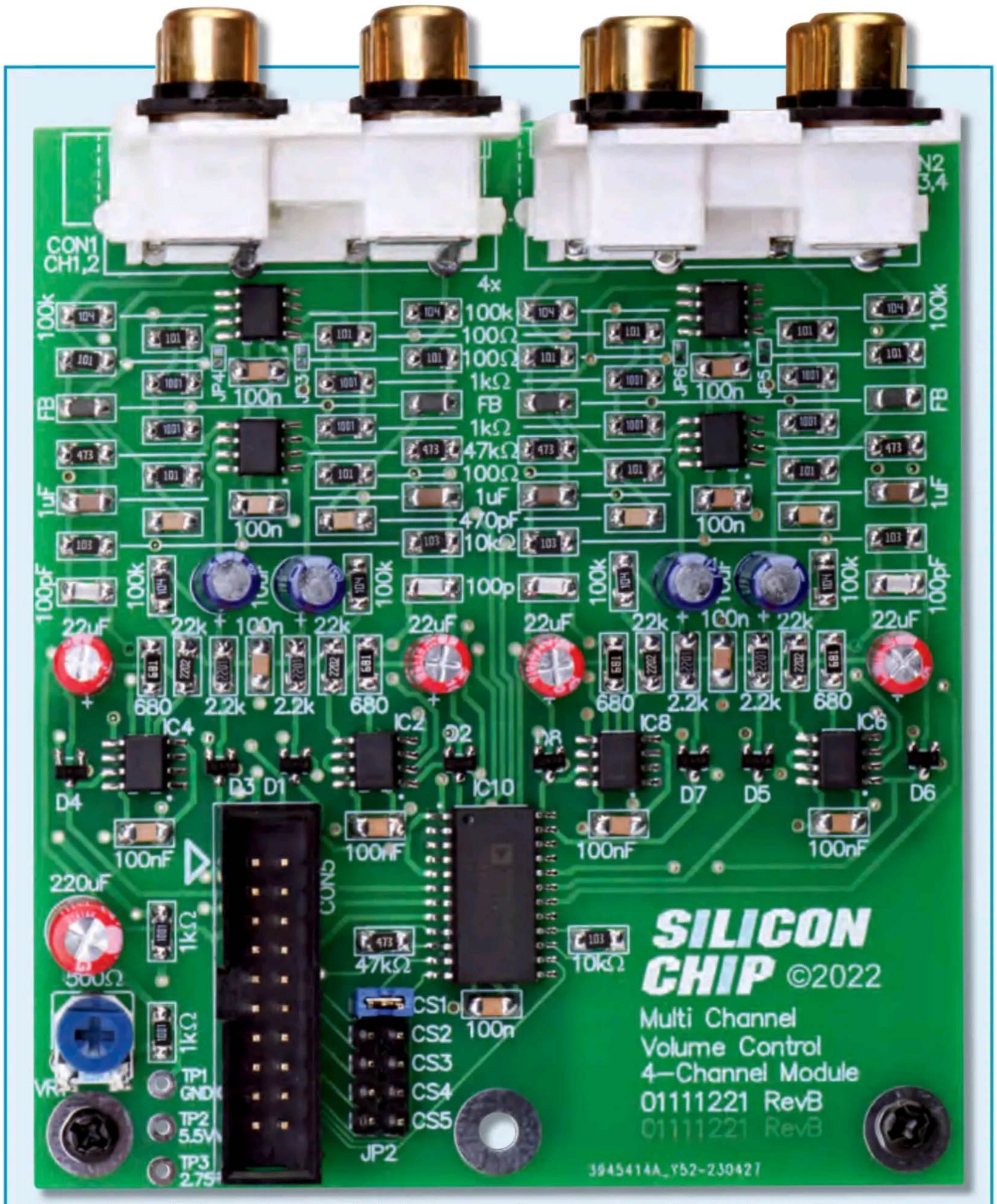

+12V
+5.5V
+2.75V
680Ω
10kΩ
100pF
100nF
IC2a
IC4b
IC6a
IC8b
B1
B2
B3
B4
D2 BAT54S
D4 BAT54S
D6 BAT54S
D8 BAT54S
2.2µF
1kΩ
TP3
VR1
500Ω
220µF
10V
JP3*
JP4*
JP5*
JP6*
47kΩ
IC3a
IC3b
IC7a
IC7b
−12V
100Ω
CON1B
CH1 OUT
CON1d
CH2 OUT
CON2b
CH3 OUT
CON2d
CH4 OUT
*JP3 – JP6 ARE PCB JUMPERS
INPUT
A
B
W
BAXANDALL VOLUME CONTROL SCHEME (ONE CHANNEL)
OUTPUT (GAIN = 0 to >1)

Fig.6: the Volume Module circuit contains four substantially identical volume control stages, each using one channel of the AD8403 precision digital potentiometer. It receives SPI control signals over the 20-way bus via CON5.


Up to five of these Volume Modules can be connected together, allowing the volume of up to 20 channels to be controlled. The modules can be incorporated into a larger system with other parts, like amplifiers.

set accurately to maximise the signal swing and ensure symmetrical clipping if the signal level is excessive.

The Volume Module has a single AD8403 quad precision digital potentiometer (IC10) powered from the 5.5V rail bypassed by a 100nF capacitor. The potentiometer's $\overline{\text{RS}}$ (reset) pin is pulled up to the 5.5V rail by 10kΩ since we do not use this feature (which forces the potentiometers to their midpoints).

The remaining digital pins of IC10 (SCK, MOSI, MISO and $\overline{\text{SHDN}}$) connect to the corresponding lines on the control bus via CON5. The $\overline{\text{CS}}$ pin goes to five-way jumper JP2, so you can choose which of the five $\overline{\text{CS}}$ lines from the microcontroller on the Control Module will control this Volume Module. Each 'slave' module in the system is assigned a different $\overline{\text{CS}}$ line.

The $\overline{\text{SHDN}}$ pin is also connected to a 47kΩ pulldown resistor to ground. This ensures that the Volume Control is muted until the microcontroller initialises and decides otherwise. The microcontroller can also drive the pin low to enforce muting, helping to eliminate noises during startup.

The remainder of the Volume Module consists of four practically identical analog sections, each using one of the 10kΩ potentiometers internal to IC10. This combination of four potentiometers (on each Volume Module) and five CS lines gives us the 20-channel limit of the Multi-Channel Volume Control.

Analog circuitry

We will describe the operation of the first channel only, as they all work the same. All 16 op amps (in eight IC packages) are high-performance (low noise and distortion) LM833 types powered from the ±12V rails. Each op amp has a 100nF bypass capacitor.

A 100kΩ resistor biases the input from the RCA socket to ground so it doesn't float if disconnected. The 100Ω resistors, 470pF capacitor and ferrite bead before the first op amp stage protect the inputs from excessive voltage swing and filter out RF noise and ultrasonic frequencies.

The first op amp stage is a unity-gain buffer to provide a high input impedance and low source impedance for the subsequent stages. The output signal is AC-coupled and biased to the 2.75V rail.

Dual diode D1 clamps the signal if it happens to deviate below the 0V rail or above the 5.5V rail, with the 2.2kΩ resistor limiting the current that flows in this case. The 0V to 5.5V span has been chosen as the widest that the AD8403 can handle in normal operation.

The 2.2kΩ resistor also forms a divider with the 10kΩ of the digital potentiometer, meaning that signals up to 2.3V RMS (or 6.5V peak-to-peak) are accepted at the input without clipping via D1.

The following two op amp stages implement the Baxandall volume control. The first stage is a unity buffer while also being a type of mixer, while the second stage is an inverting amplifier with a gain of 14.7. One effect of this is that this stage has its output polarity inverted with respect to its input.

The potentiometer is connected between the input ('A' end) and output ('B' end) of these two stages, providing logarithmic gain changes despite the circuit using a linear potentiometer. Another dual diode at the other end of the digital potentiometer's track to protects it from excessive voltages from the output of the gain stage.

When the $\overline{\text{SHDN}}$ signal is asserted (low), the digital potentiometer disconnects the A end of each potentiometer and connects the wiper to the B end of the track; this means that output is fully muted.

The potentiometer exhibits a small amount of resistance at the end of each wiper, so even at the extreme 'low' setting, a small amount of signal will pass through to the output unless the $\overline{\text{SHDN}}$ signal is used.

Another capacitor and resistor bias the signal back to its original ground reference, and the final op amp stage for each channel applies more gain to allow us to get an output swing of up to 2.5V RMS, despite the 5.5V peak-to-peak limitation imposed by the digital pot ICs. A 100Ω resistor between the output and RCA socket isolates the op amp from capacitive loads.

There is an option to use a jumper to bypass the last op amp stage to reduce the gain, saving a handful of components in the process.

An important feature is that the four digital potentiometers can all be set independently, allowing the channels to have different levels if needed to maintain audio balance.

SILICON CHIP
Volume
0.0dB
MUTE ON
Press to mute
Multi Channel Volume Control

This optional OLED Module provides a more compact status display and a rotary volume control. If you don't want to use the touchscreen, you could leave it off and make this as the full interface (and with an IR receiver wired back to the Control Module).

OLED Module

The OLED Module provides an optional set of compact, tactile controls. Its circuit is very simple (see Fig.7) as it has been designed so that the PCB also forms a front panel with its components on the back. This Module is intended to be mounted like a bezel over a cutout in an existing panel.

It connects to the ribbon cable bus via its CON12, using the same 20-way IDC box header as the other modules. On this Module, only the SPI control lines, the five CS lines, and 3.3V power and ground are connected; its operation is entirely digital and does not require the other supply rails.

A 14-pin 8-bit PIC16F15224 microcontroller (IC11) is connected to the SPI bus as a slave, with its $\overline{CS}$ line picked from the five on the bus by JP7. JP7 uses solder shorting pads rather than pin headers and a jumper to keep this module compact.

A further three pins of IC11 are connected to a rotary encoder, which provides the user input. Two of these pins are for the quadrature encoder and are either pulled to ground by the contacts in the rotary encoder or pulled up by the 10kΩ resistors.

The pushbutton in the rotary encoder is connected to a similar arrangement, with each of these three pins also having a 100nF capacitor to ground to help filter out contact bounce.

Two further pins from IC11 are also used to drive the OLED via an I²C serial interface. This OLED shows the Volume Control's state as it is updated.

The typical arrangement for 8-bit PICs is also present, consisting of a supply bypass capacitor between 3.3V and ground with a pullup to the $\overline{MCLR}$ pin. These three pins and the PGC and PGD pins used for in-circuit programming are also taken to the in-circuit programming header, CON13.

Volume control range limitations

Using a quad 8-bit digital potentiometer (IC10), rather than a purpose-designed volume control IC, helps keep costs down. However, it imposes a limitation on the effective volume control range.

A dedicated volume control IC might give a range of 100dB or more, but an 8-bit digital potentiometer only has

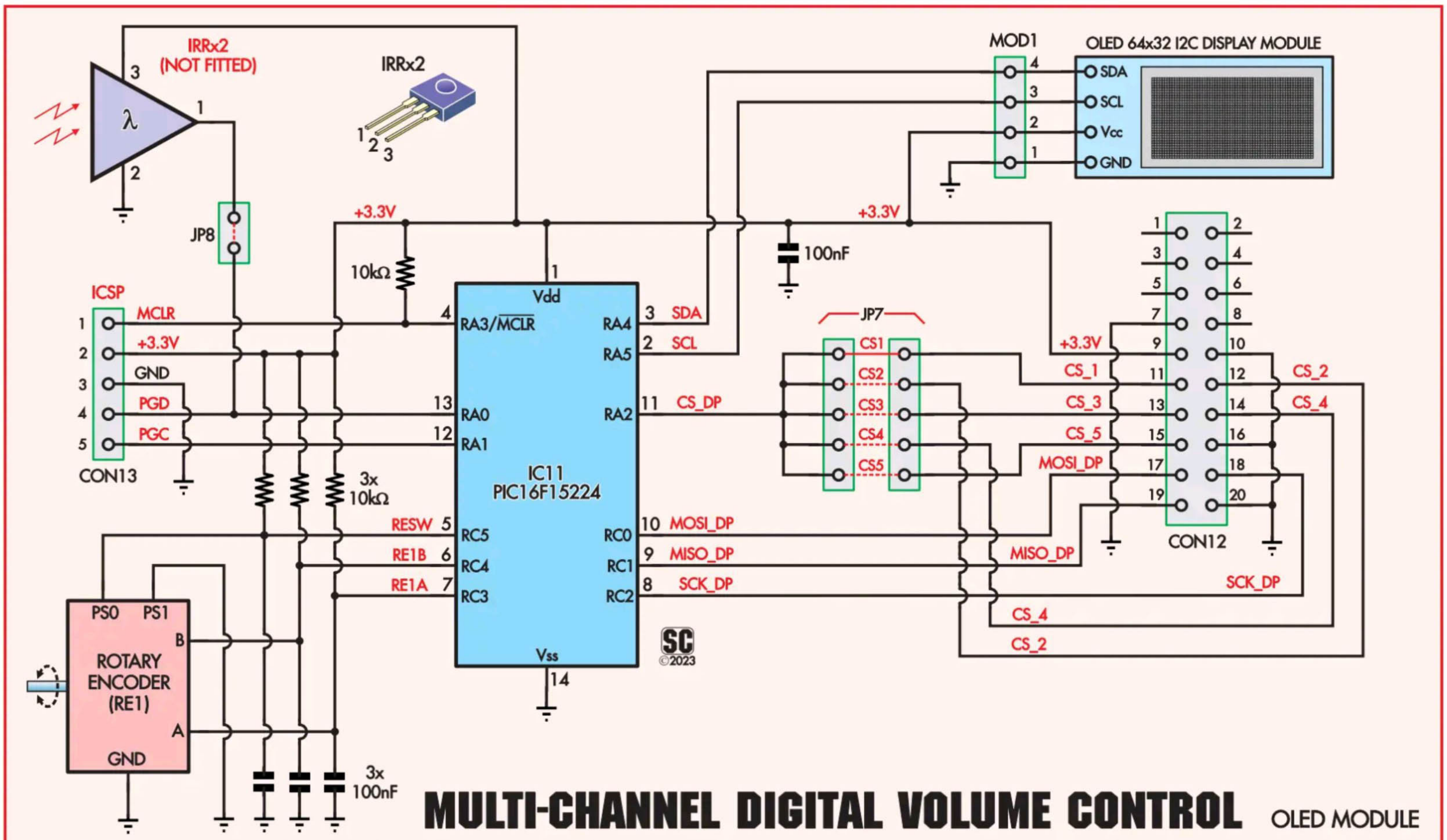

IRRx2
(NOT FITTED)
λ
3
1
2
IRRx2
1
2
3
JP8
+3.3V
10kΩ
ICSP
1 MCLR
2 +3.3V
3 GND
4 PGD
5 PGC
CON13
1
Vdd
4 RA3/MCLR
RA4 3 SDA
RA5 2 SCL
13 RA0
12 RA1
RA2 11 CS_DP
IC11
PIC16F15224
3x 10kΩ
RESW 5 RC5
RE1B 6 RC4
RE1A 7 RC3
RC0 10 MOSI_DP
RC1 9 MISO_DP
RC2 8 SCK_DP
Vss
14
PS0
PS1
B
ROTARY ENCODER (RE1)
A
GND
3x 100nF
MOD1
OLED 64x32 I2C DISPLAY MODULE
4 SDA
3 SCL
2 Vcc
1 GND
+3.3V
100nF
JP7
CS1
CS2
CS3
CS4
CS5
+3.3V
CS_1
CS_3
CS_5
MOSI_DP
MISO_DP
CS_2
CS_4
SCK_DP
CS_4
CS_2
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
CON12
SC ©2023
MULTI-CHANNEL DIGITAL VOLUME CONTROL
OLED MODULE

Fig.7: the OLED Module circuit; it is a simple microcontroller-based board with an OLED and rotary encoder. Since it only requires the 3.3V rail and the SPI bus, only 13 of the 20 pins on CON12 are connected

256 steps, meaning it has an effective volume control range of about 60dB.

That's enough for most applications. Still, you should check that the highest output level makes sense, or you might end up without good control at lower volume settings.

The default configuration has a maximum gain of 16dB, giving a full-scale output of around 2V RMS from an input signal close to 300mV RMS. That's well below line level, which is typically more like 775mV RMS.

In this configuration, the lowest signal output level with a 1V RMS input before muting is 4mV RMS or -48dB. The steps above that are 9mV (-41dB), 15mV (-36.5dB), 21mV (-33.5dB), 27mV (-31.4dB), 33mV (-29.6dB) and steps of about 1dB or less from there up.

While -48dB is 64dB below the maximum output level, the steps are pretty large until around -30dB, giving a useful control range of about 46dB.

A variation of, say, 10dB between different input signal sources will reduce the effective volume control range to 36dB. That gives a 4000:1 ratio between the highest and lowest power output with decent control; if maximum volume results in 100W from your amplifier, you will have fine control down to just 25mW.

That's certainly good enough, but the more the maximum possible gain is above what you need, the more apparent the steps at lower volume levels will become.

The system's overall gain is set with resistors, so you can easily adjust it at the construction stage. Practically speaking, if your power amplifier will reach full power with less than 2V RMS (as many will, and all your input-signals are at least line-level), we suggest you omit the final 6dB op amp gain stages. That will give you 6dB more room at the lower end of the volume range.

Parts List – Multi-Channel Volume Control

1 Control and Power Supply Module (see below)
1-5 Volume Modules (each handles four channels; see below)
1 OLED Module (**optional**; see below)
1m length of 20-way ribbon cable (cut to suit application)
1 universal IR remote control (**optional**; see text)
1 12V AC single winding or 24V AC centre-tapped transformer and appropriate wiring/fusing
RCA cables to interface to existing hardware
Other mounting hardware to suit your application

Control and Power Supply Module

1 double-sided PCB coded 01111222, 87 × 50mm
1 2.8in LCD touchscreen with ILI9341 controller & SPI interface
1 2.1mm or 2.5mm DC jack socket (CON7; **optional**)
1 3-way 5mm/5.08mm pitch screw terminal block (CON8; **optional**)
1 14-way 0.1in female header (CON9; for LCD touchscreen)
1 5-way right-angle pin header (CON10; **optional**, for ICSP)
1 20-way box header and IDC inline plug (CON11) ***OR***
1 20-way IDC transition header (CON11)
1 500Ω mini top-adjust trimpot (VR2)
1 3-pin infrared receiver, 38kHz (IRRx1) **[eg, TSOP4138, TSOP33438 etc]**
9 M3 × 5mm panhead machine screws
4 M3 × 12mm tapped spacers
1 M3 hex nut and washer (for mounting REG1)

Semiconductors

1 W04M bridge rectifier (BR1)
1 PIC16F18146-I/SO microcontroller programmed with 0111122B.HEX, wide SOIC-20 (IC9)
1 IRLML2244TRPBF or SSM3J372R 20V 1A+ logic-level P-channel Mosfet, SOT-23 (Q1)
1 2N7002 60V 115mA N-channel Mosfet, SOT-23 (Q2)
1 7805 +5V 1A linear regulator, TO-220 (REG1)
1 78L12 +12V 100mA linear regulator, TO-92 (REG2)
1 MCP1700-3.3 3.3V 250mA linear regulator, SOT-23 (REG3)
1 79L12 -12V 100mA linear regulator, TO-92 (REG4)
1 LM317L 100mA adjustable linear regulator, TO-92 (REG5)
1 5.6V 1W zener diode (ZD1)

Capacitors

2 1000µF 25V electrolytic
2 220µF 10V electrolytic
4 100µF 16V electrolytic
1 1µF 10V X7R ceramic, SMD M3216/1206 size
2 100nF 50V X7R ceramic, SMD M3216/1206 size

Resistors ***(all SMD M3216/1206 size 1% except as noted)***

2 10kΩ 2 1kΩ 1 910Ω 1 560Ω
1 110Ω 1 100Ω 1 22Ω 5% 5W axial

Control firmware

We've chosen an 8-bit microcontroller for the Control Module as the requirements are not too burdensome. Although it is driving an LCD panel, the user interface is not complex, with only a single screen configuration needed (no menus etc).

The LCD screen is overlaid by a touch panel, which the micro scans for user input.

We know of two otherwise interchangeable versions of this panel with touch panels rotated by 180° compared to each other. Our workaround is to display the user control buttons in the bottom half of the screen. Touches in the top half of the panel are assumed to correspond to touches on the bottom half of the rotated display, so either should work with no changes.

In retrospect, we might have chosen a more powerful (and faster) microcontroller, such as the 16-bit PIC24FJ256GA702. At the time of writing, they are not dissimilar in price, although the SSOP version of the PIC24FJ256GA702 is somewhat more tricky to solder than the SOIC part we are using.

The lesser flash memory available on the PIC16F18146 meant that we needed to compress the large font that's necessary to provide a clear and legible display.

The PIC16F18146 microcontroller we used for this project can store

Multi-channel Volume Control, part 1

Volume Module
1 double-sided PCB coded 01111221, 82 × 94mm
2 quad right-angle RCA socket assemblies (CON1, CON2) **[Altronics P0214]**
1 20-way box header and IDC inline plug (CON5) ***OR***
1 20-way IDC transition header (CON5)
4 SMD ferrite beads, M3216/1206 size (FB1-FB4) **[Fair-Rite 2512066017Y1]**
1 5×2 pin header and a jumper shunt (JP2)
1 500Ω mini top-adjust trimpot (VR1)
8 M3 × 6mm panhead machine screws
4 M3 × 12mmm tapped spacers
(or other mounting hardware to suit the application)

Semiconductors
8 BAT54S dual series schottky diodes, SOT-23 (D1-D8)
8(6) LM833 low-noise dual op amps, SOIC-8 (IC1-IC8)
1 AD8403ARZ10 quad precision digital potentiometer, SOIC-28 (IC10)

Capacitors *(all SMD M3216/1206 size unless noted)*
1 220µF 10V electrolytic
4 22µF 16V electrolytic
4 10µF 16V electrolytic
4 2.2µF 25V SMA size SMD tantalum ◆
11(9) 100nF 50V X7R
4 470pF 50V C0G/NP0
4 100pF 50V C0G/NP0

Resistors *(all SMD M3216/1206 size 1%)*
8 100kΩ   5 47kΩ   4 22kΩ   5 10kΩ
4 2.2kΩ   10(2) 1kΩ   4 680Ω   12 100Ω

n ***numbers in brackets refer to requirements if the last op amp gain stage is omitted***
◆ ***not recommended but 22µF 4V+ X5R/X7R ceramics in M3216/1206 can be substituted (see panel on "Lessons learned during development")***

OLED Module
1 double-sided PCB coded 01111223, 51 × 76mm
1 0.96in I²C OLED module (MOD1)
1 PIC16F15224-I/SL microcontroller programmed with 0111122C.HEX, SOIC-14 (IC11)
1 pulse-type rotary encoder with 18 tooth spline shaft (RE1)
1 knob to suit RE1
1 20-way SMD box header or 20-way dual-row SMD header (CON12) ●
1 20-way IDC inline plug
1 5-way pin header (CON13; **optional**, for ICSP)
4 M3 screws, washers and nuts to suit mounting requirements
4 100nF 50V X7R M3216/1206 size SMD ceramic capacitors
4 10kΩ M3216/1206 size SMD 1% resistors
several short pieces of solid wire (eg, component lead offcuts)
● ***can be made by cutting 10 rows from a long DIL SMD header***

Control Module Kit
SC6793 (~£31): also comes with 1m of ribbon cable

Volume Module Kit
SC6794 (~£34): includes all the listed parts

OLED Module Kit
SC6795 (~£15): includes all the listed parts

Each kit includes all the parts listed under each module in the parts list. The only other items needed are a case, power supply and remote control.

around 16kB of font data in its flash memory, so you can see how important careful managing font data is. We were very close to running out of flash memory before we started looking into compression.

Font compression

To display text on a graphical screen requires some form of font data to encode the 'glyphs' (character representations) to show. The glyphs usually correspond to a subset of the ASCII character set, perhaps with minor alterations to suit the project, such as including the degrees (°) symbol instead of some other less-used character.

The font data format we typically use is widely known. It consists of two header bytes describing the width and height of the font in pixels, followed by another two bytes denoting the first ASCII character code point (eg, 32 for a space is typical) and the number of characters within the font.

This is followed by bits of pixel data in bitmap form for each character within the font. Since each byte can hold eight pixels, a small 8×8 pixel font of 95 characters (the full ASCII set) takes up 764 bytes.

Fig.a (in the panel overleaf) shows how such a font is translated from bitmap data into the corresponding C code. Other languages, such as MMBasic, use a similar format adapted to the syntax of the specific language.

A 16×24 pixel font, our typical choice for legible text on a typical 3.5in LCD panel, uses around six times as many bytes or just over 4kB. Larger fonts are possible by upscaling smaller fonts, but the result does not look as good.

The flash memory of these 8-bit PICs uses 14-bit words. They are often described as having 28kB of flash memory, but it is arranged as 16k x 14-bit words rather than 28k x 8-bit bytes.

The ideal font size for the large dB display in this project is 44×60 pixels; that would take 31,354 bytes to store as a full ASCII font, which obviously wouldn't fit in the PIC16F18146.

We can reduce the space required by only encoding the characters we need. For example, truncating the 12×16 font we use for buttons and smaller text to only include from the space character up to the capital letters brings its size down from 2284 bytes to 1420 bytes.

Similarly, we reduced the font used for the dB display to the digits 0-9, a blank space, a negative sign, a decimal point (or full stop) and the 'd' and 'B' characters. That takes it down to 4954 bytes before we apply RLE compression.

RLE compression

RLE stands for run length encoding and takes advantage of repetitive values in data; in this case, runs of the same pixel colour. RLE has existed in displays and computing for at least 50 years. RLE is used in the JPEG image standard (although it is only part of the compression used there). It was also used by fax machines (remember them?).

For our implementation, rather than storing bitmap data, the data encodes a run of pixels of the same state (on/off). The top bit indicates whether the pixels are on or off (ie, add 128 for on pixels), while the remaining seven lower bits encode the number of consecutive pixels with that property.

Fig.b shows a glyph encoded using our RLE strategy. This small font is not a great example for this sort of compression, as the resulting data has ballooned from eight bytes up to 25 bytes.

RLE could be seen to encode how often the pixel state changes on each line. So characters such as "1" should encode better than "0" and, indeed, the RLE data only comes to 15 bytes for "1" in this particular font.

The space character for this font is encoded as a single byte of value 64, meaning all 64 pixels are off.

This variability in glyph size is accounted for by adding a header specifying the number of bytes before each group of RLE data. It's easy to write code to step through the data jumping forward by the header's byte count until we reach the glyph we need to display; we then have the count of the number of bytes we need to decode.

The RLE-compressed font we ended up with only uses 2013 bytes (including all header data) compared to the 4954 bytes for the uncompressed version.

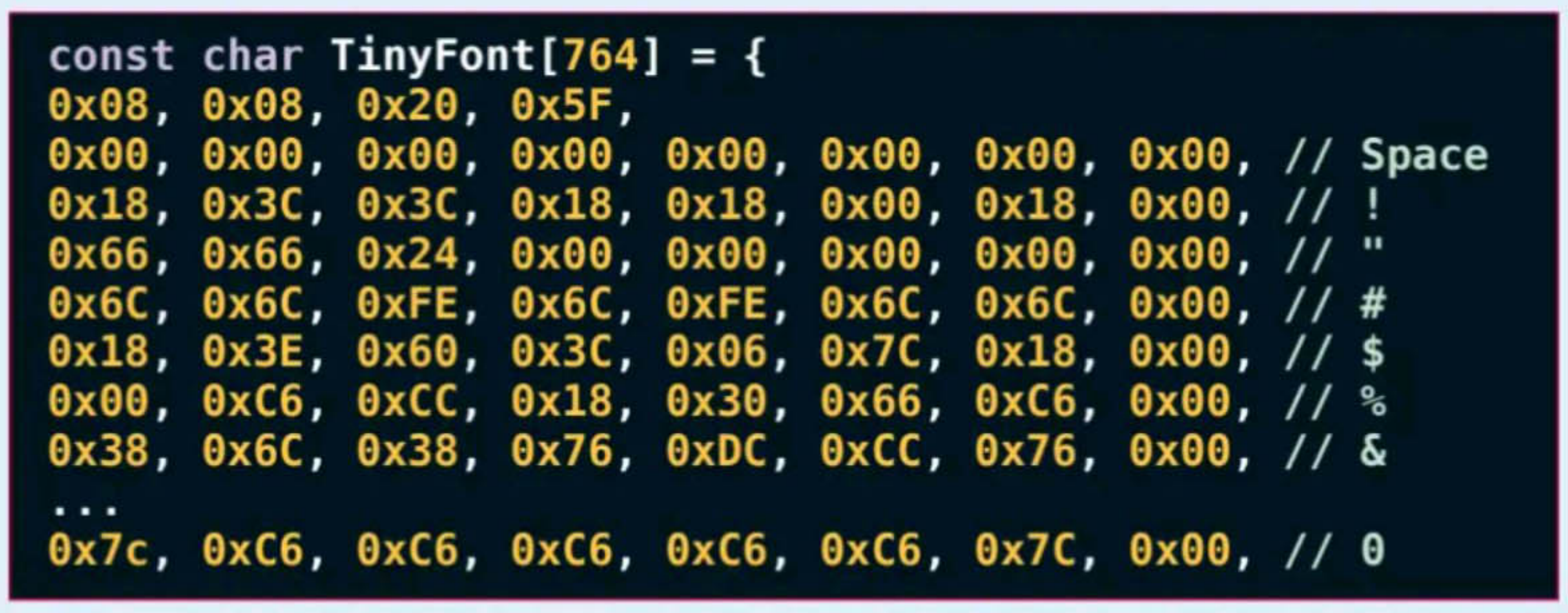
As an extreme example, the space character for this large font (which consists of 2640 black pixels) takes up 330 bytes uncompressed, but only 22 bytes after compression. Another extreme, the '0' character, takes up the same 330 bytes uncompressed but only 196 bytes when compressed, a 40% saving.

For the numerals 0-9 and space, the RLE encoding provides a notable saving for fonts as small as 16×24 pixels. Different font subsets, including letters like M and W, will not compress as well as they include shorter runs.

Other advantages

The microcontroller can copy the RLE-compressed data to the screen faster than bitmap data. There is less data to be read from flash in the RLE case and the decoding is simpler too.

The uncompressed bitmap data needs to be decoded one bit at a time. Each individual bit has its value checked, then the appropriate colour pixel is written to the LCD. For the RLE data, a group of consecutive pixels is decoded and can be efficiently sent to the LCD in a tight loop.



```
const char TinyFont[764] = {
0x08, 0x08, 0x20, 0x5F,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, // Space
0x18, 0x3C, 0x3C, 0x18, 0x18, 0x00, 0x18, 0x00, // !
0x66, 0x66, 0x24, 0x00, 0x00, 0x00, 0x00, 0x00, // "
0x6C, 0x6C, 0xFE, 0x6C, 0xFE, 0x6C, 0x6C, 0x00, // #
0x18, 0x3E, 0x60, 0x3C, 0x06, 0x7C, 0x18, 0x00, // $
0x00, 0xC6, 0xCC, 0x18, 0x30, 0x66, 0xC6, 0x00, // %
0x38, 0x6C, 0x38, 0x76, 0xDC, 0xCC, 0x76, 0x00, // &
...
0x7c, 0xC6, 0xC6, 0xC6, 0xC6, 0xC6, 0x7C, 0x00, // 0
```

Fig.a: bitmap font data is encoded with a '1' bit indicating a pixel is set to the foreground colour or a '0' bit if the pixel is the background colour. The last line shows the bitmap data for the number 0.

Similarly, the code to show the RLE-encoded font is smaller than its bitmap equivalent, about half the size in program flash memory. The bitmap routine also calls another function to perform a multiplication; there would be further savings if the same routine were not needed by other code.

Being able to update the display more quickly is a clear upside. We suspect the effect would not be so pronounced with a faster microcontroller, but it would still be present.

OLED fonts

We have not used RLE fonts in the firmware for the OLED Module. Firstly, these displays are monochrome, so they do not require individual pixel colours to be written. Instead, they expect data to arrive in blocks of eight bits at a time, which maps to eight pixels on the screen. In other words, they natively work with bitmap data.

Also, the OLED has a much lower resolution, so it does not need to display very large fonts as often. Given that larger fonts benefit more from RLE, there is less incentive to apply it to the smaller fonts.

Bitmaps image compression

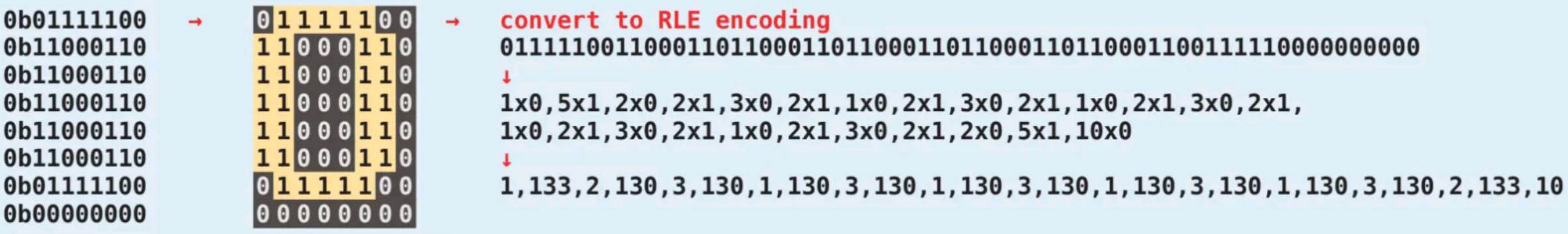
The Silicon Chip logo shown on the LCD screen is a bitmap and is stored similarly to the bitmap fonts. So, we experimented with a different encoding that stores a run of pixels (from 0 to 15) in each nibble of a byte. The top nibble is assumed to be the background colour, while the lower nibble is the foreground colour.

The code to decode this data is similar in speed and program flash memory usage, and we found this algorithm offered about 40% compression on the Silicon Chip logo. So we used this encoding for the logos and icons that are displayed.

Conclusion

RLE encoding a larger font gives superior image quality than upscaling a smaller font, is faster and can use a similar amount of flash or even less. So it seems like the way to go. The only downside is the extra complexity in the initial encoding.

Some font examples can be found at:
www.rinkydinkelectronics.com/r_fonts.php
Download the FontTweak Font editing program from:
www.c-com.com.au/MMedit.htm



```
0b01111100  →  01111100  →  convert to RLE encoding
0b11000110     11000110     0111110011000110110001101100011011000110110001100111110000000000
0b11000110     11000110     ↓
0b11000110     11000110     1x0,5x1,2x0,2x1,3x0,2x1,1x0,2x1,3x0,2x1,1x0,2x1,3x0,2x1,
0b11000110     11000110     1x0,2x1,3x0,2x1,1x0,2x1,3x0,2x1,2x0,5x1,10x0
0b11000110     11000110     ↓
0b01111100     01111100     1,133,2,130,3,130,1,130,3,130,1,130,3,130,1,130,3,130,1,130,3,130,2,133,10
0b00000000     00000000
```

Fig.b: mapping of the pixels to bitmap data is from top left to bottom right in horizontal rows. For an eight-pixel wide font, each row maps to one byte of data. With RLE encoding, each run of same-coloured pixels maps to a byte of data, so one byte can encode up to 127 pixels. It is much more effective for fonts that have more glyphs.

Lessons learned during development

We solved two significant but subtle problems in developing this design. The first was that, to save components and simplify setup, we generated the +2.75V rail on the Power Supply & Control board and fed it via the ribbon cable to the Volume Modules.

The problem with this was that any tiny amount of noise or ripple picked up in the ribbon cable ends up getting injected into the signals because this is a virtual ground rail. Adding extra capacitance to ground for this rail on each Volume Module didn't fix it. The only way to get acceptable performance was to move the +2.75V rail generation circuitry onto the Volume Modules.

The other problem we ran into was that we accidentally used 1µF X7R multilayer ceramic capacitors to couple the signal to the last op amp stage (eg, from pin 1 of IC2a to pin 3 of IC3a). This type of capacitor simply isn't very linear and the result was a significant rise in distortion below 200Hz, shown by the dashed portions of the curves in Fig.1.

There are two solutions to this. Our preferred solution is to switch to using 2.2µF tantalum capacitors, which luckily are available in the same size (the SMA tantalum case is basically the same dimensions as M3216/1206 ceramic chip capacitors). Being electrolytic capacitors, these are not as linear as say plastic film types, but significantly more linear than X7R ceramics.

Unlike ceramic capacitors, tantalum capacitors are polarised. There is 2.75V between these points, so the capacitors are orientated with the positive leads to the op amp pin 1 outputs.

As a less-desirable alternative, X7R multi-layer ceramic capacitors can still be used but with a significantly higher value; at least 10µF, and ideally 22µF or more. That pushes the distortion down so instead of starting below 200Hz, it starts below 20Hz, which is in the inaudible part of the frequency range.

There usually won't be much signal below 20Hz; our concern is that, if there is, the resulting distortion harmonics could be in the audible range. Hence our preference for the tantalum capacitors.

The compression technique used is called run-length encoding (RLE). Rather than storing simple bitmaps for the characters to show on the display, RLE stores a run of pixels as its count and on or off state (hence run-length encoding).

The resulting font data is less than half its original size. That is helpful when the font uses up nearly a third of the available flash memory. Another advantage is that the encoding is easy to decompress for display, making screen updates faster! We've gone into this in greater detail in a panel, for those interested in that sort of thing.

Infrared remote control

The IR decoding routine is designed to receive signals encoded with the NEC protocol. It uses pulse position encoding with 38kHz modulation; the IR receiver demodulates this carrier, so the microcontroller only needs to decode the pulse encoding.

The microcontroller uses a timer/counter to measure the pulse lengths and thus extract the necessary data. The NEC protocol transmits 32 data bits, including a device byte, its inverse, a command code and the command code's inverse. The inverses allow the microcontroller to reject corrupted codes.

The Control Module can be programmed to accept different device and command codes, so it can be used with fixed or programmable IR remote controls. We'll get more into those details later.

The Control Module communicates over the SPI bus on the ribbon cable. If an OLED module is present, data is sent out from the Control Module to update its display; the Control Module also receives data back if there has been activity on the rotary encoder or its button.

The micro on the control module signals the digital potentiometers to set a new volume level whenever the volume is changed. If the mute function is activated, the volume is ramped down as low as possible, after which the $\overline{\text{SHDN}}$ pin is pulled low to mute the audio fully.

Unmuting is simply the reverse. The $\overline{\text{SHDN}}$ pin goes high, then the volume level ramps up. The same process occurs when the unit is powered on, avoiding clicks and pops.

Otherwise, the main loop updates the display when necessary and reads input from the touch panel.

We had provision for an IR receiver on the OLED Module, but since the Control Module is mandatory and already has an IR receiver, we have not fitted it to our prototypes and there is no support for it in the firmware.

OLED module firmware

The microcontroller on the OLED Module monitors the rotary encoder and button for action, sends and receives data to and from the Control Module and updates the OLED screen as required.

It acts as an SPI slave device, meaning it must be ready to respond whenever the Control Module wants to communicate. For simplicity, we designed the communication between these two modules to only use a single byte in each direction.

The Master sends out the volume level (in steps of 0.5dB) relative to a value of 128. A value of zero means that the mute is active. Thus, the OLED Module doesn't need to retain any state data; it simply updates its display whenever data is received.

The data that the OLED Module sends back to the Control Module is in a similar format. The OLED module counts the number of rotary encoder steps (forward or backward) that have accumulated and sends that to the Control Module, offset from a value of 128.

The offset helps avoid receiving spurious commands if no OLED Module is connected. The data is designed not to use the values of 0 or 255 (00000000 and 11111111 in binary) as might occur if the data line was pulled up or pulled down permanently.

A special value of 51 (00110011 in binary) indicates a press of the rotary encoder's button. This works as a toggle, so the OLED Module does not need to know the current state and simply reports to the Control Module that the mute state needs to change.

Next month

That's all we can fit in this month's article; we will describe the construction and assembly of the Preamp in the following issue. Short-form kits for all three modules are available, so you might like to gather the parts together in preparation. PE

Teach-In 2024

Learn electronics with the ESP32

by Mike Tooley

Part 9 – Putting it all together

In the last part of our *Teach-In* series, we showed how a low-cost GPS module can be used with an ESP32 to provide location, navigation and time data. Our Practical Project involved the construction of a simple GPS location fixer using an ESP32 development board, a GPS module and a 20 × 4 LCD screen, while Coding Workshop looked at passing parameters into user-defined functions.

In this final part, we begin by introducing keypad interfacing and our Practical Project deals with the construction of a simple passcode-protected entry control system. Coding Workshop deals with files and file handling, and shows how you can easily store data using an SD card interface. Checkpoint brings the series to a conclusion with another opportunity to test your knowledge of the ESP32.

The learning objectives for the ninth part of our series are to know how to:

- interface with a matrix keypad
- set and use passcodes
- read data from and write data to an SD (or microSD) card
- use basic file handling techniques

About Teach-In

Our latest *Teach-In* series is about using the popular ESP32 module as a basis for learning electronics and coding. We will be making no assumptions about your coding ability or your previous experience of electronics. If you know one but not the other, you have come to the right place. On the other hand, if you happen to be a complete newbie there's no need to worry because the series will take a progressive hands-on approach. There will be plenty of time to build up your knowledge and plenty of opportunity to test things out along the way.

We've not included too much basic theory because this can be easily found elsewhere, including several of our previous *Teach-In* series, see:

https://bit.ly/pe-ti
https://bit.ly/pe-ti-bundle

Each month, there'll be projects and challenges to help you check and develop your understanding of the topics covered.

Keypads

Many simple interfacing projects require only a few buttons and switches for user input. Where that's the case, you can use some of the ESP32's digital I/O pins configured as inputs as described previously in this Teach-In.

However, if you need to deal with quite a few buttons or switches, things can get a little complicated, and you can easily run out of digital inputs. That's certainly the case if you need to interface a keypad. Fortunately, those are usually arranged in the form of a matrix of rows and columns, requiring fewer of your precious I/O lines, leaving you more for other devices such as motors, relays and displays.

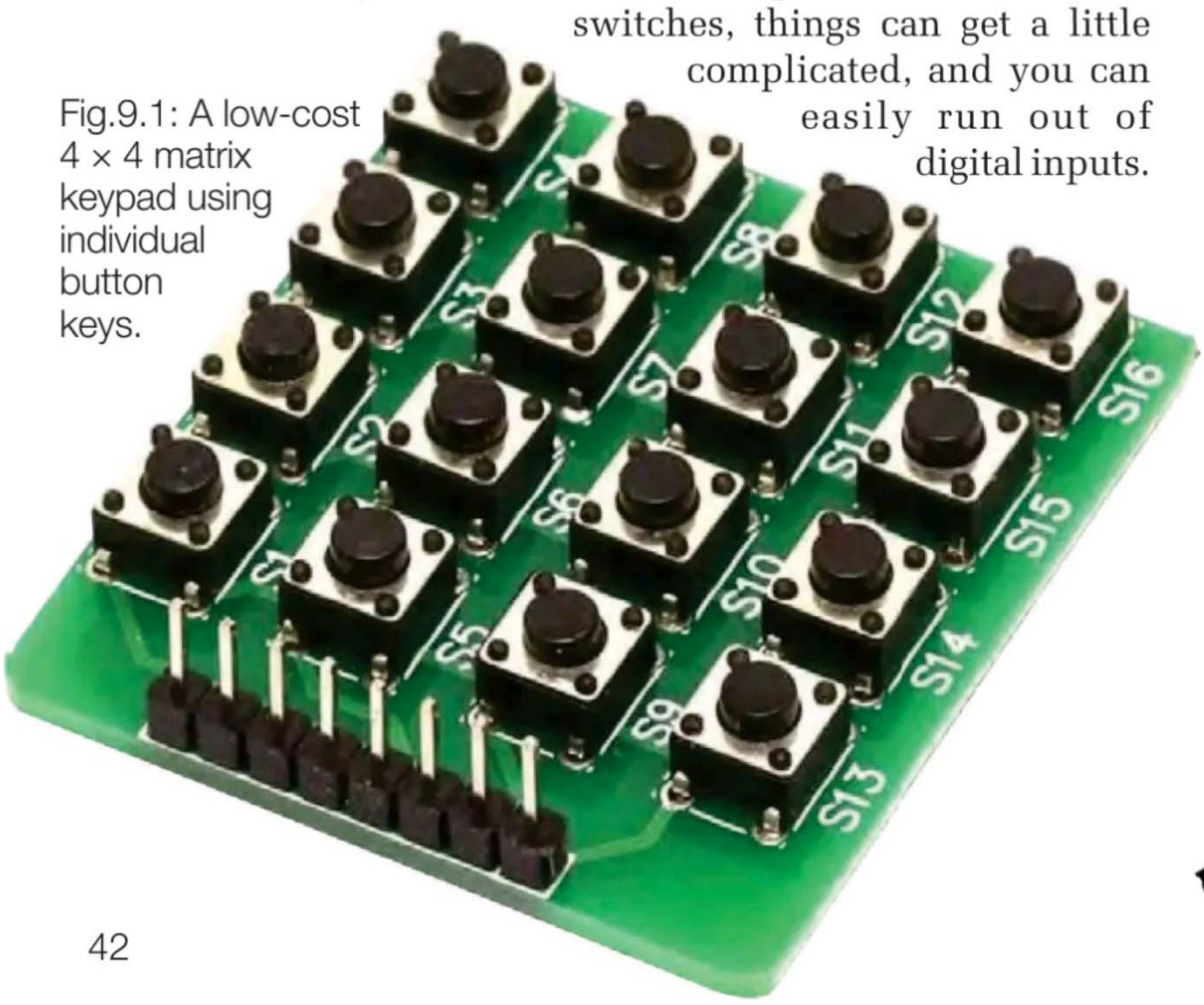


Fig.9.1: A low-cost 4 × 4 matrix keypad using individual button keys.

Figs.9.1 & 9.2 show some typical low-cost matrix keypads suitable for use with the ESP32. They are both arranged as a 4 × 4 matrix, but the keypad in Fig.9.1 is based on 16 individual pushbutton keys while that shown in Fig.9.2 uses a membrane. Fig.9.3 shows their circuit arrangements, whilst Fig.9.4 shows pin numbering for the keypad's 8-way ribbon connector in Fig.9.2.

A standard 4 × 4 matrix keypad will give you a total of 16 keys to play with: 0 to 9 and A to D with additional * and # characters. The keys are placed at the intersection of the rows and columns so that, for example, when the A key is pressed, the top row is linked to the right-most column and

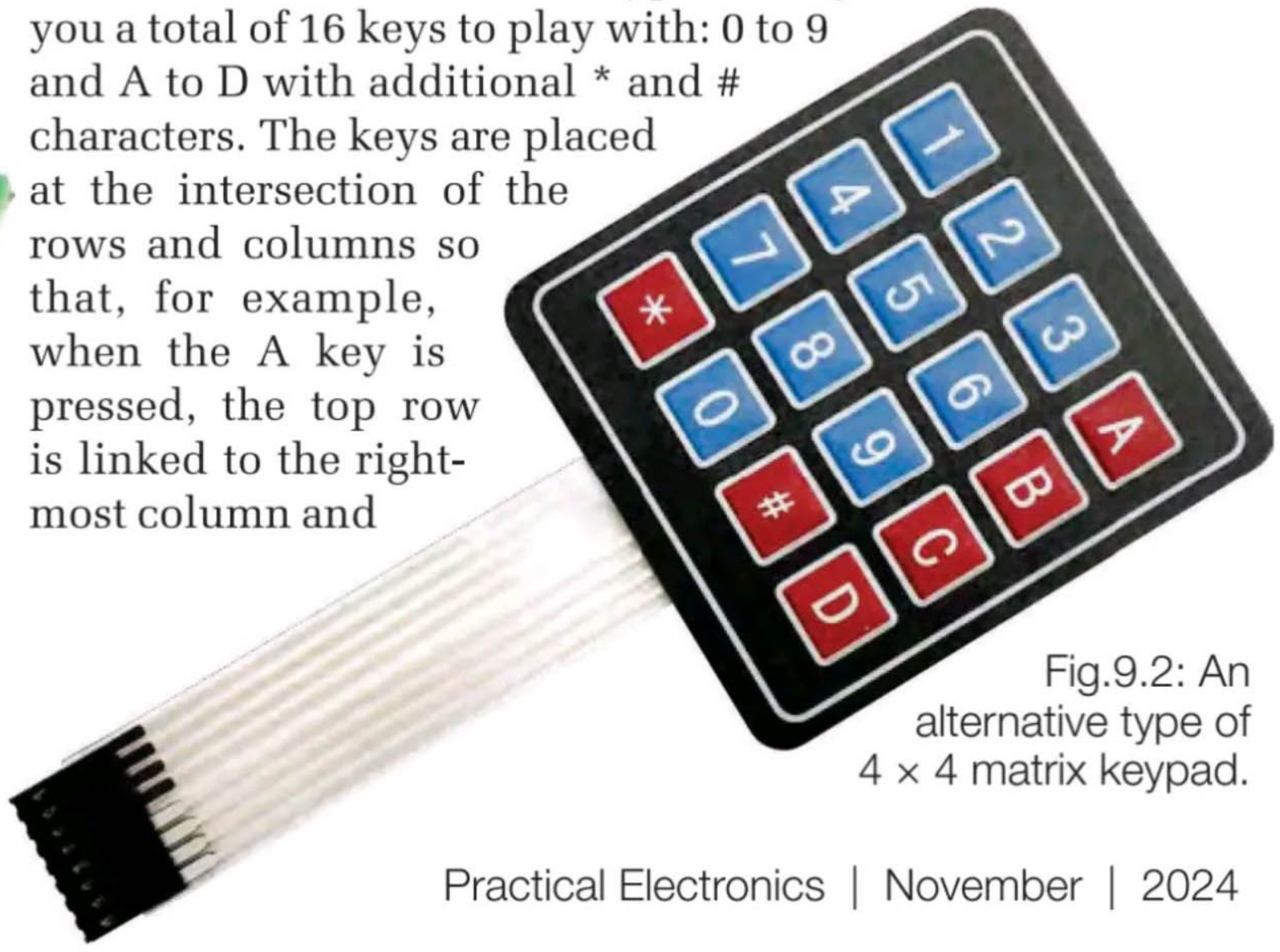


Fig.9.2: An alternative type of 4 × 4 matrix keypad.

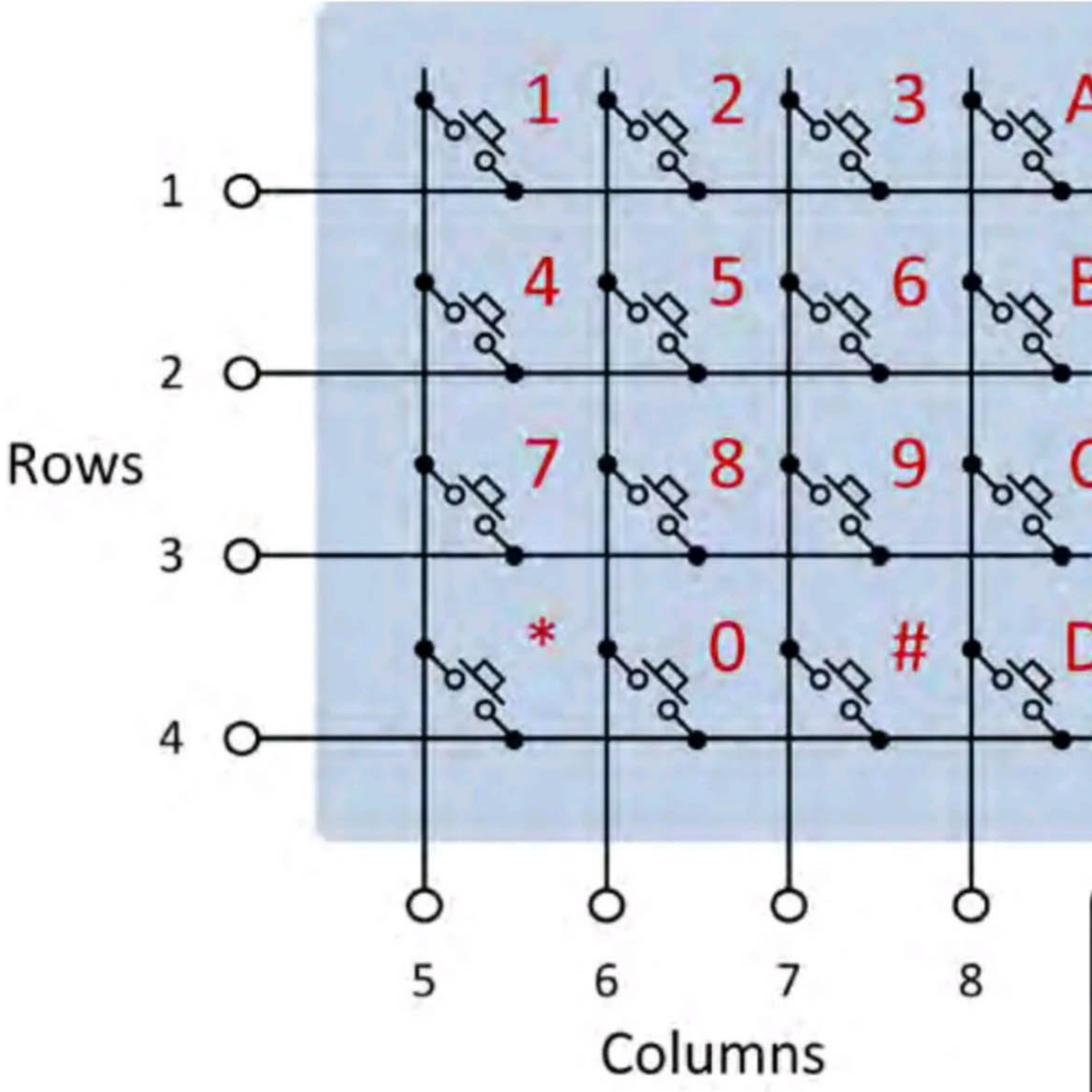


Fig.9.3: The wiring diagram of the 4 × 4 matrix keypad.

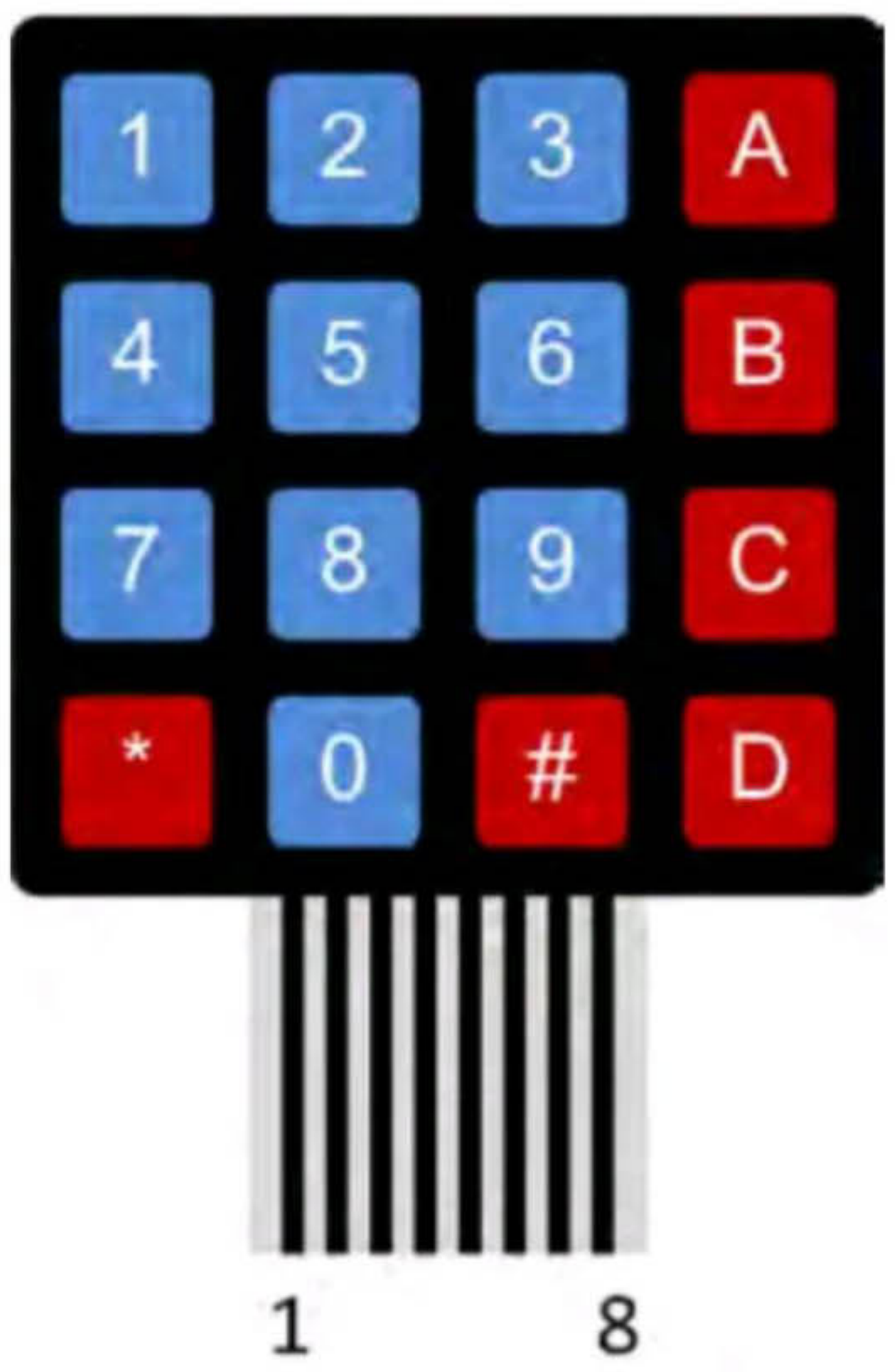
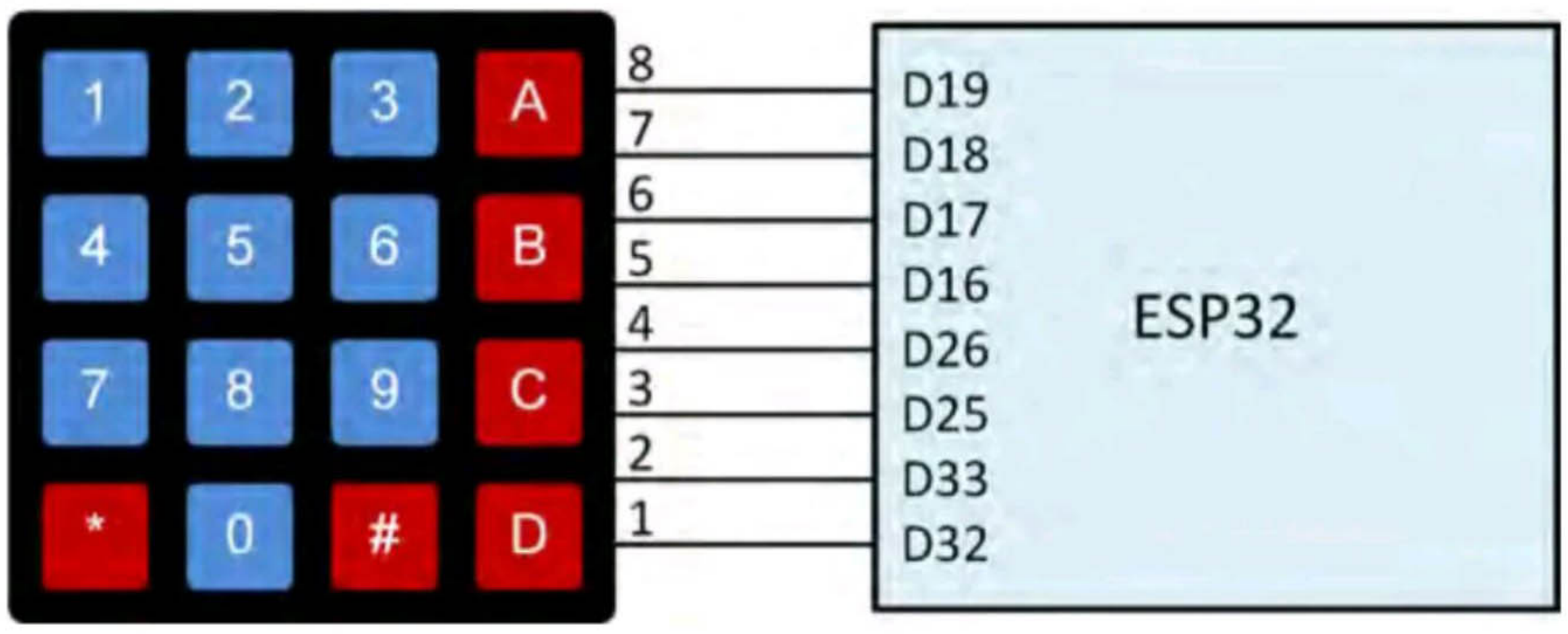


Fig.9.4: The ribbon cable numbering for the 4 × 4 matrix keypad.

Fig.9.5: Connecting a keypad to an ESP32.



consequently, pins 1 and 8 will be connected. On the other hand, if the key marked 9 is pressed, pins 3 and 7 will be joined.

All we need to detect a keypress and decide which key is being pressed is code that will address the rows in turn and look for a signal returned on one of the columns. So, if pin 1 is taken high and a high level is detected on pin 8, the A key is being pressed. Similarly, if pin 3 is taken high and a high is detected on pin 7, the 9 key has been pressed.

Note that in these cases, you would need some sort of pull-down on the input pins (either internal to the microcontroller or in the form of an external resistor) to ensure the pins stay low unless being pulled high via the keypad switches.

Depending on your code, the key scanning could be done in a few places. If your program has an 'inner loop' that executes contiuously and loops frequently enough (say, at least 10 times per second), you could add a call to a function to scan the keypad within that inner loop.

Alternatively, a better approach would be to use a timer to trigger an interrupt frequently (say, around 50 times per second) that could read the state of the inputs and note whether any key has been pressed (for the main loop to act on later). After that, it would update the state of the outputs to scan the next row or column. That method would work even if the code is busy with other tasks.

If that all sounds a bit complicated, fortunately, there's no need to write your own keypad scanning code. Someone else has already performed that arduous task for you!

Check it out

To check out the matrix keypad, all you need is an eight-wire connection from the ESP32 development board to the keypad, as shown in Figs.9.5 & 9.6.

The problem of connecting the keypad ribbon cable to the ESP32 is easily solved using a coloured ribbon multi-jumper lead with male and female connectors at the ends, as shown in Fig.9.7.

When you've completed the wiring, you will need to download, compile and execute Listing 9.1. But before you do that, you will need to locate and install the keypad library.

You can use the Library Manager to install the keypad library file. Open the Library Manager then click on "Tools", followed by "Manage Libraries".

The Library Manager will open, and you will be presented with a list of installed libraries as well as those that are

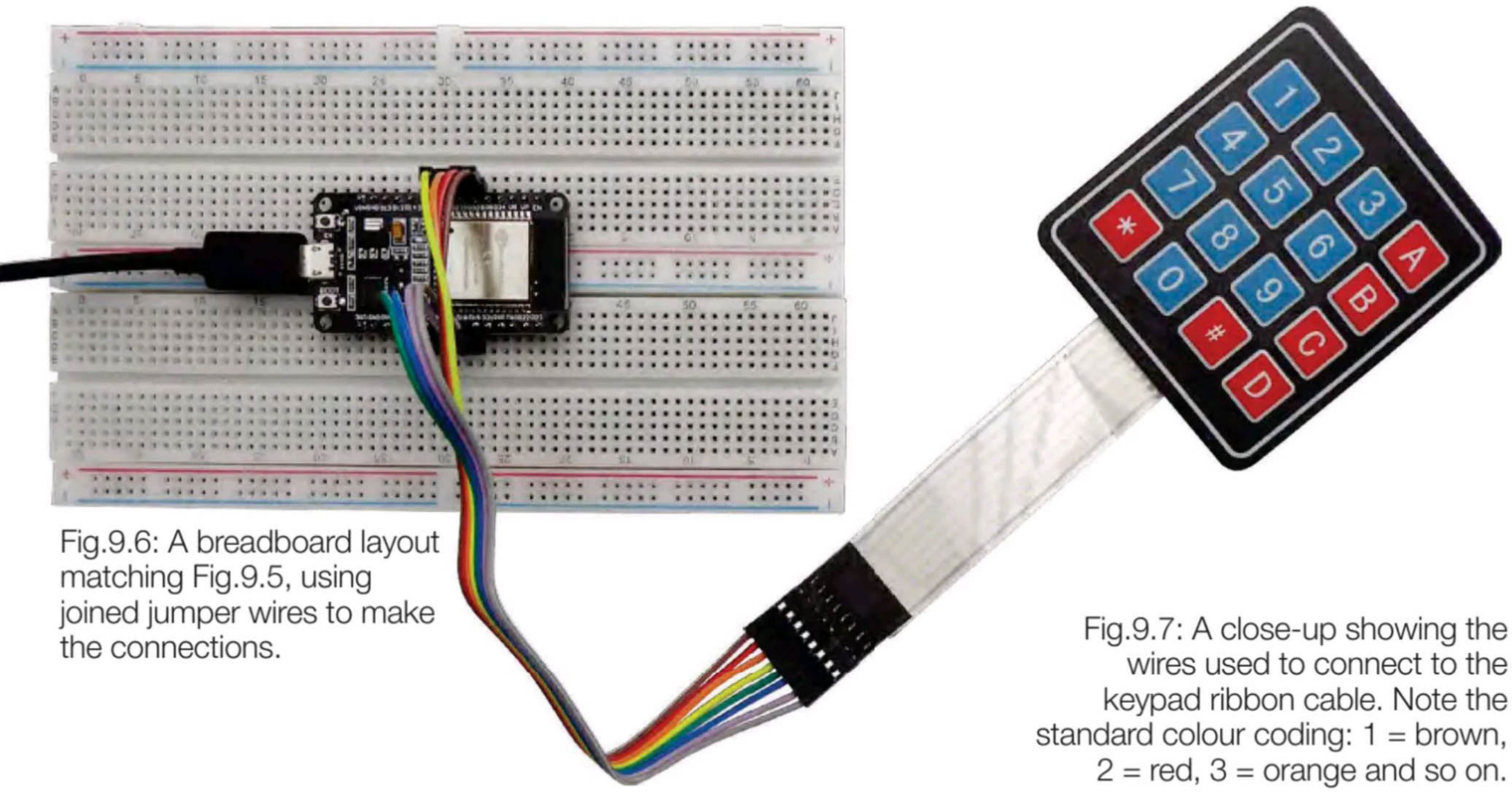


Fig.9.6: A breadboard layout matching Fig.9.5, using joined jumper wires to make the connections.

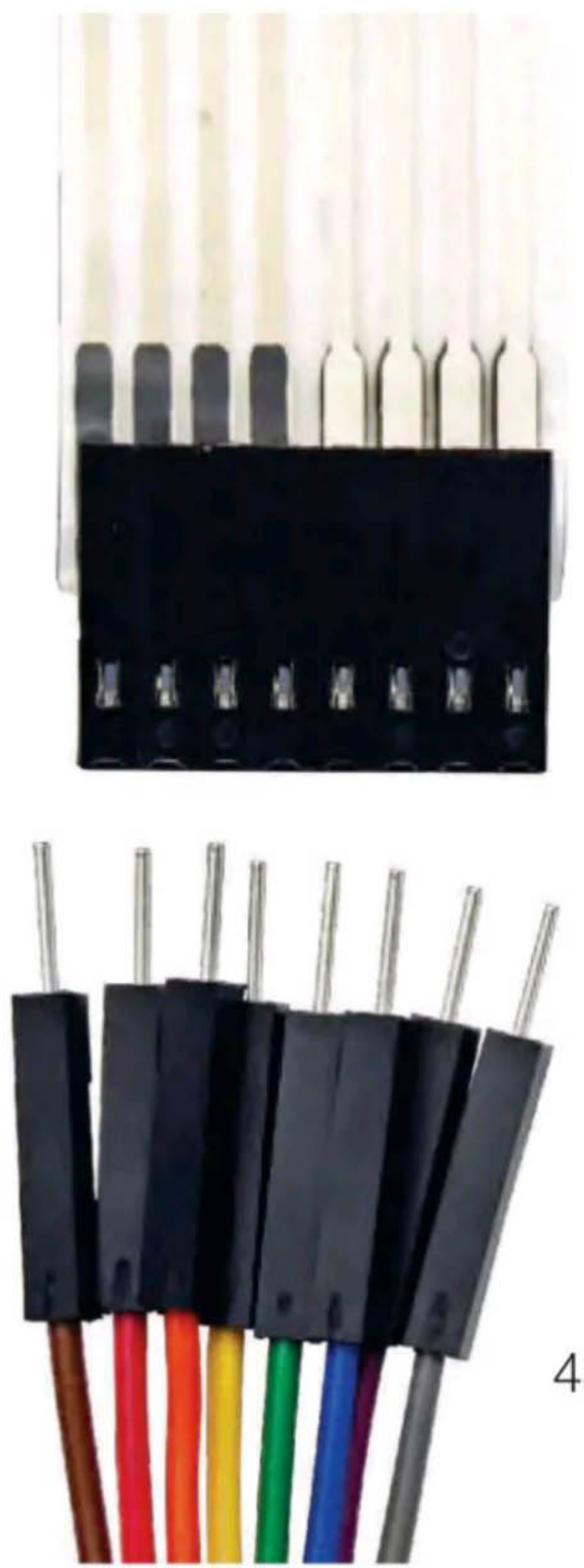
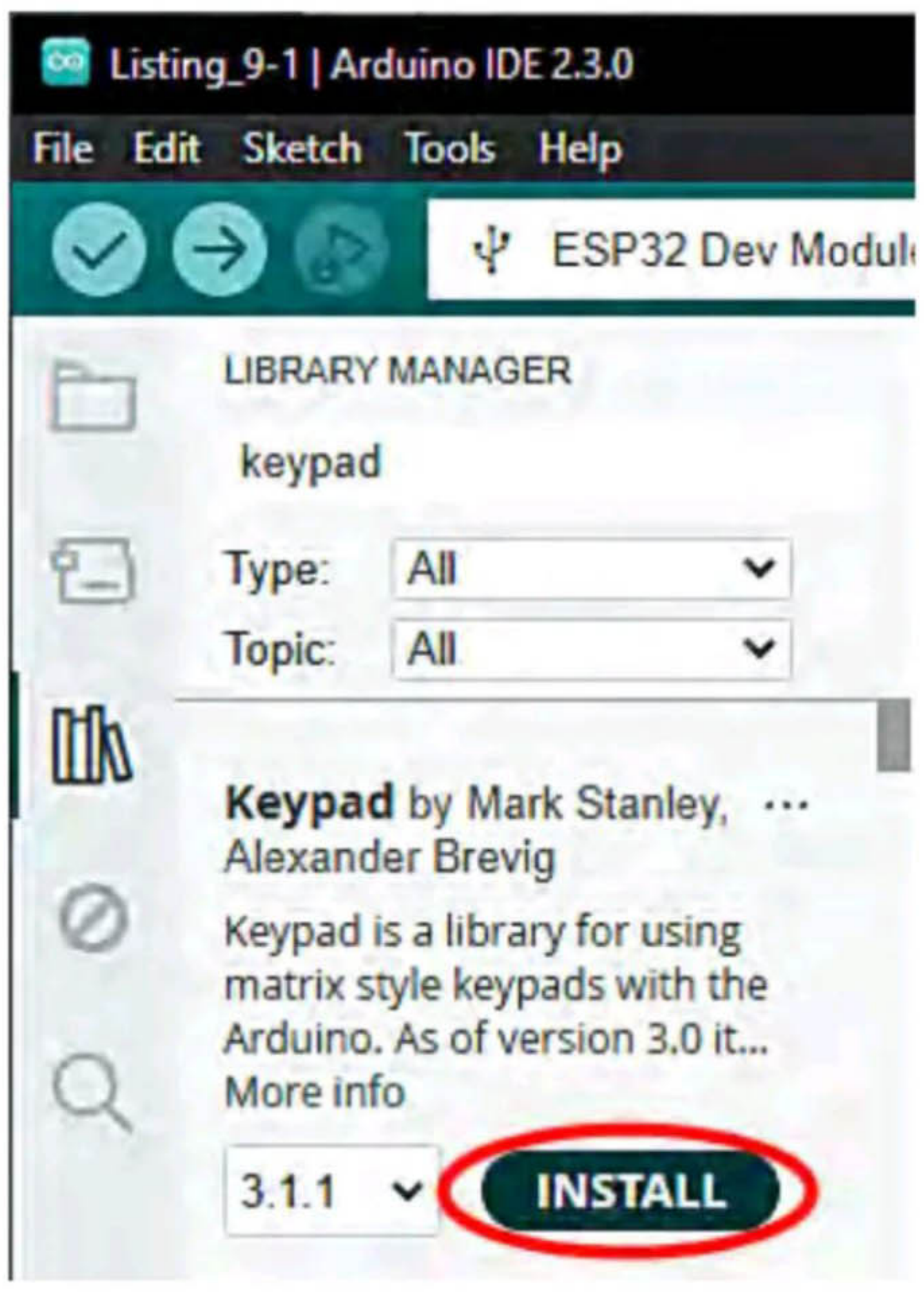


Fig.9.7: A close-up showing the wires used to connect to the keypad ribbon cable. Note the standard colour coding: 1 = brown, 2 = red, 3 = orange and so on.

Listing 9.1 – Keypad test code

```
#include <Keypad.h>    // Include the keypad library

const byte ROWS = 4;  // The keypad has four rows
const byte COLS = 4;  // and four columns
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'},
};   // Use an array to hold the key values

byte colPins[ROWS] = {16,17,18,19}; // Define the I/O
byte rowPins[COLS] = {32,33,25,26}; // pin numbers

Keypad keypad =
Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
  Serial.begin(9600);  // Initialize the Serial Monitor
}

void loop() {
  char key = keypad.getKey();  // Let's check the keypad
  if (key != NO_KEY){          // and if a key was
    Serial.println(key);       // pressed, print it!
  }
}
```



keypad

Type: All

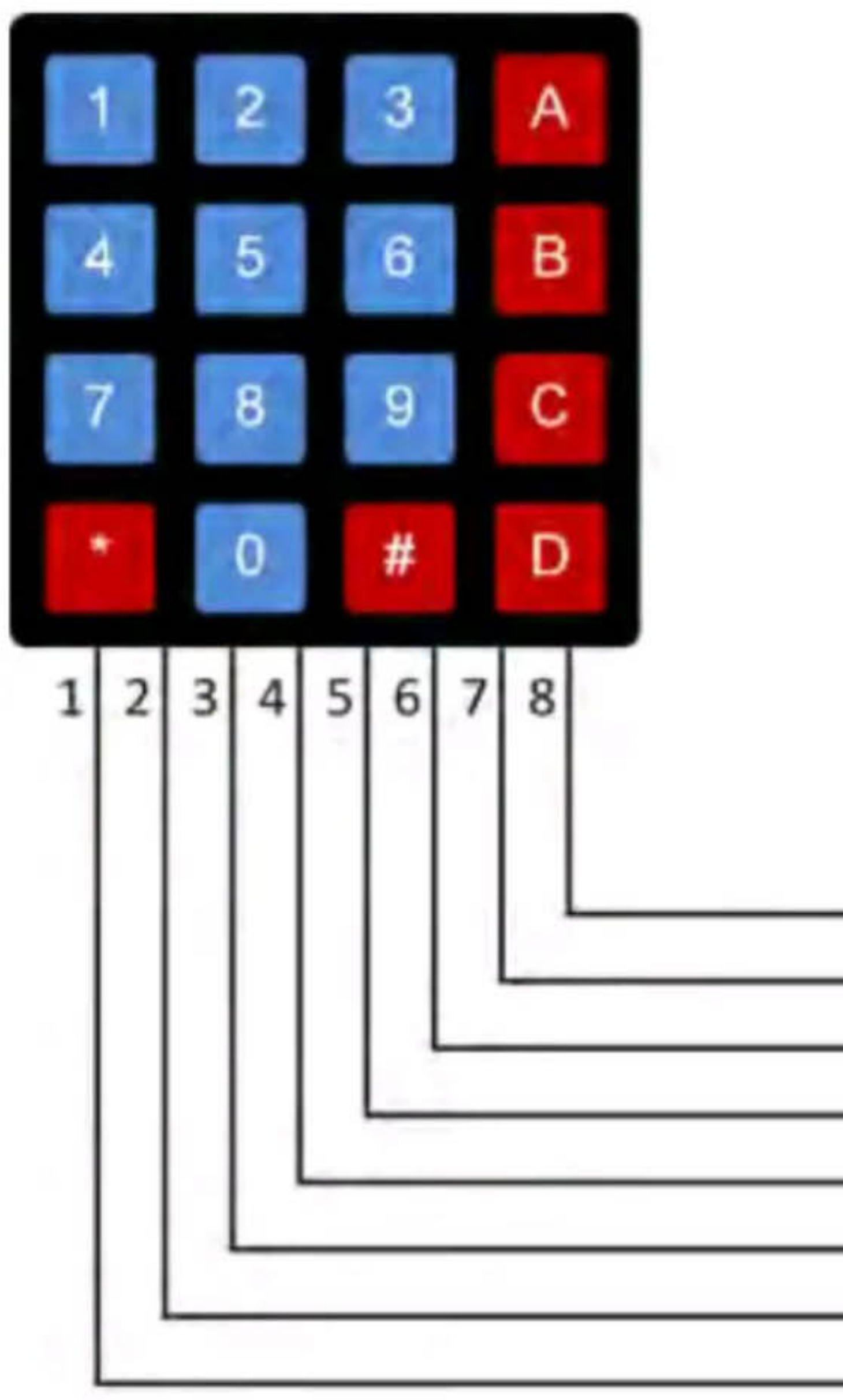
Topic: All

Keypad by Mark Stanley, Alexander Brevig

Keypad is a library for using matrix style keypads with the Arduino. As of version 3.0 it... More info

3.1.1

Fig.9.8: Using the Library Manager to install the Keypad library.



available for installation. To shorten the list, enter "keypad" in the search field and then scroll down to find the keypad library, as shown in Fig.9.8.

Having located the library, you just need to click on "INSTALL" and wait for the IDE to install it. Depending on the speed of your Internet connection, downloading might take a few minutes.

When the installation is complete, you should notice that the "INSTALL" button has been replaced by a "REMOVE" button. You should also notice the messages that appear in the status window (see Fig.9.9). You can now close the Library Manager.

To include the keypad (and before you make use of the keypad object), you will need to add this line at the beginning of the code:

```
#include <Keypad.h>
```

The keypad test code shown in Listing 9.1 is available for download from the PE website at https://pemag.au/link/abyx. When the code has been compiled and executed, you will be able to press the keys and note the characters displayed via the Serial Monitor (see Fig.9.10).

Practical project

The ESP32 makes an ideal candidate for use in a wide range of simple security and alarm systems. Our Practical project this month shows how you can implement a simple but effective entry system. You will need the following items:

- an ESP32 development board on a breadboard
- a 4 × 4 keypad
- a backlit 16 × 2 alphanumeric LCD screen
- a 5V piezoelectric sounder
- a 3.3/5V relay module

The complete circuit of the simple entry system is shown in Fig.9.11. It uses low-cost, commonly available components. The 16 × 2 LCD screen and 16-key keypad are widely available from several suppliers, including those based in the Far East. The relay module is like those that we've used earlier in this Teach-In series: a 3.3/5V type with suitably rated contacts.

Connection to the door solenoid is provided by a three-way terminal block fitted to the relay module. The breadboard layout is shown in Fig.9.12.

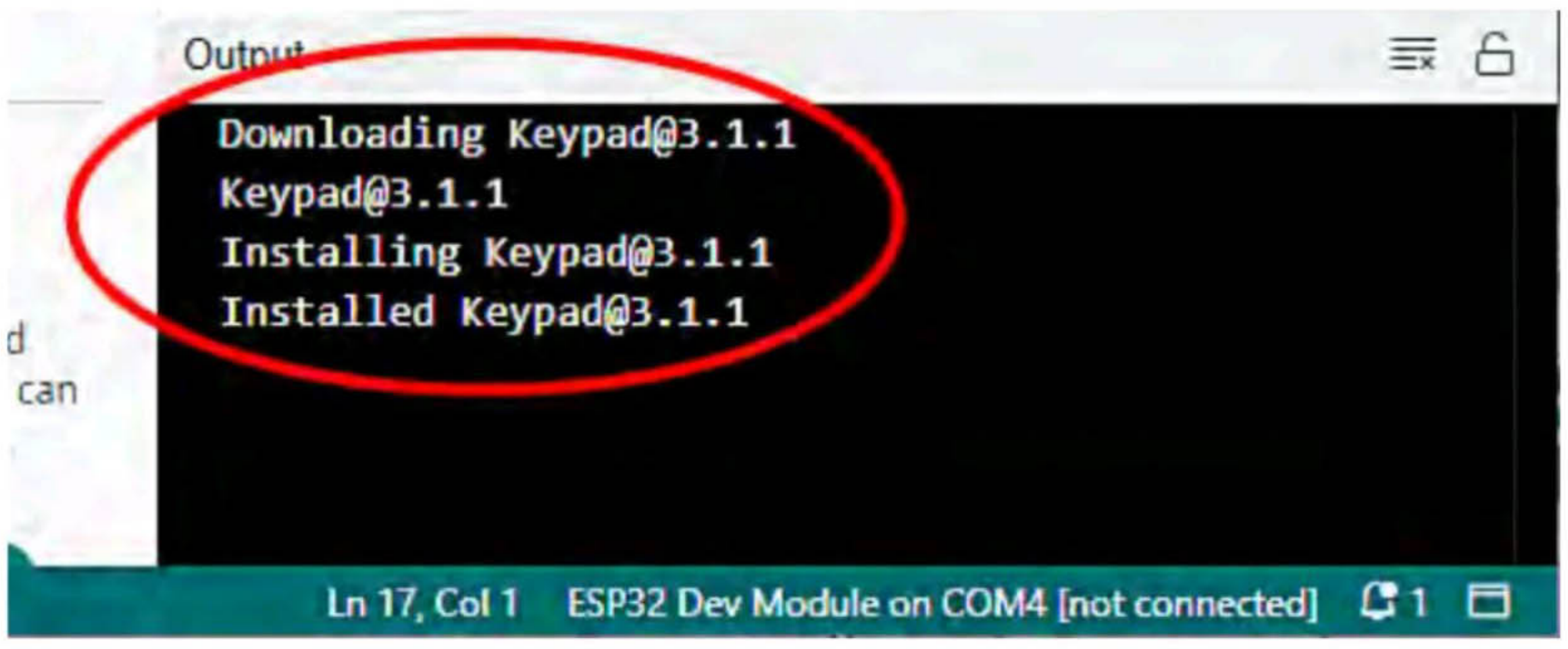
In practise, the user enters a passcode consisting of a string of alphanumeric characters, followed by the '#' key. A beep from the piezoelectric sounder provides confirmation that the user has entered a character on the keypad.

When the correct passcode has been entered, the LED will illuminate and the relay will operate for a time that can be preset within the code. The passcode for access is also preset in the code.

Listing 9.2 shows the complete code for the simple ESP32-based entry system. As usual, we've added comments to help you understand what's going on.

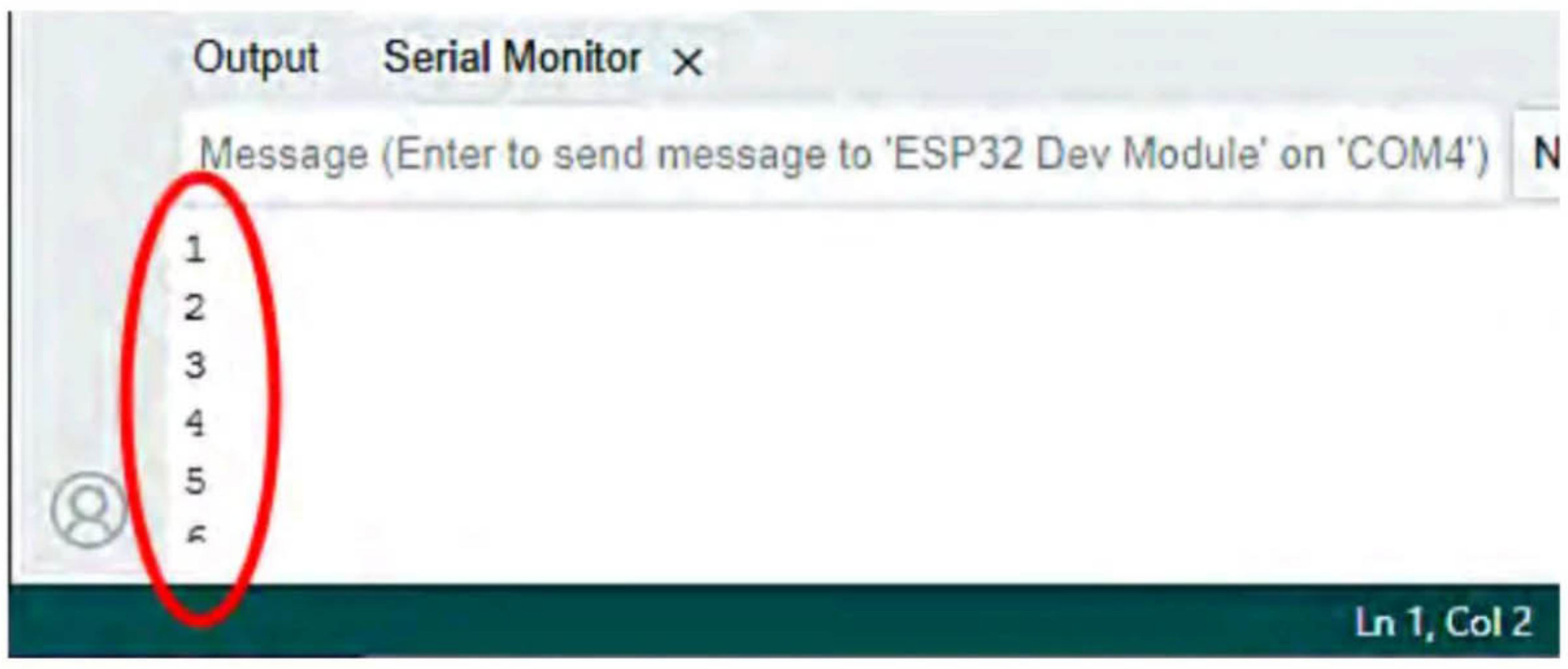
The code requires three libraries; LiquidCrystal.h, keypad.h and Password.h. The first two of these provide functions that facilitate the use of the LCD display and the matrix keypad, whilst the third deals with setting and checking the passcode string.

You could develop some of your own code to do those tasks, but there's little point in doing that if someone else



```
Downloading Keypad@3.1.1
Keypad@3.1.1
Installing Keypad@3.1.1
Installed Keypad@3.1.1
```

Fig.9.9: The messages at the bottom of the window show that the Keypad library has been successfully installed.


Output
Serial Monitor
Message (Enter to send message to 'ESP32 Dev Module' on 'COM4')
Ln 1, Col 2

```
1
2
3
4
5
```

Fig.9.10: These characters in the Serial Monitor were generated by pressing buttons on the keypad.

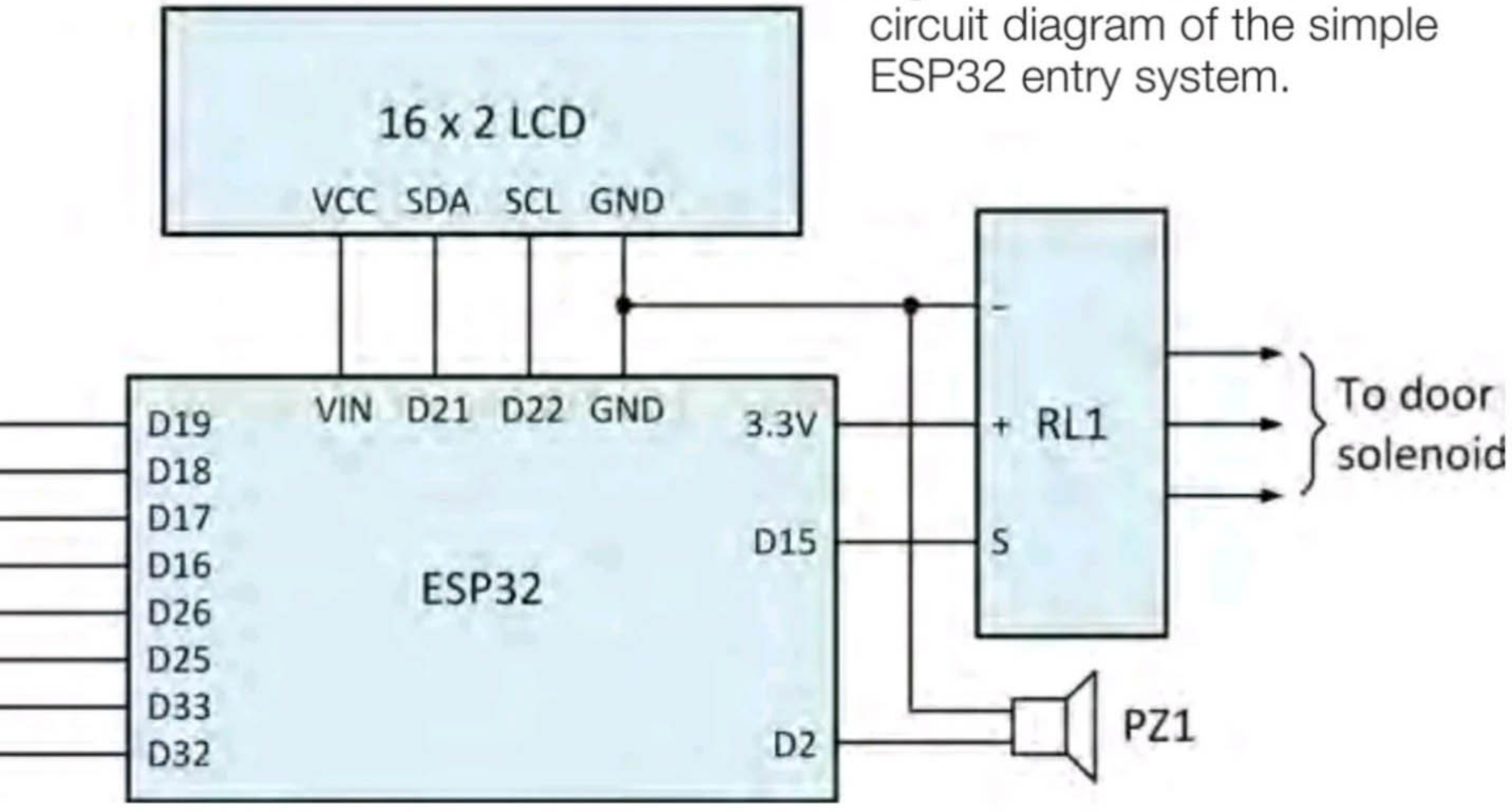


Fig.9.11: The complete circuit diagram of the simple ESP32 entry system.

has already provided a simple and elegant solution that they have shared with the rest of the community!

The password library can be downloaded and installed using the Library Manager (as explained earlier).

When you've corrected any coding errors reported by the compiler, you will be ready to execute your code. Just click on the arrow and watch the progress report. When the ESP32 performs a reset, you will see some characters appearing across the LCD for a short time, after which the display will clear and show the message, "Enter Passcode".

At this point, the ESP32 is waiting for user input from the keypad. If you then press a key, you will hear a beep from the piezoelectric sounder, which informs you that the character that you've pressed has been accepted.

If you press several further characters in succession, followed by the '#' key, the entered passcode will be checked and a message, either "Welcome" or "Invalid Passcode" will appear. If the correct passcode (ABCD) has been entered prior to the '#' key, the relay module's green LED should illuminate and the relay will operate for two seconds (sufficient to release the door). This time interval can easily be changed by modifying the value shown in the code.

As usual, there's a great deal of scope for going further with the entry control system. The most obvious enhancement would be the addition of access codes that would enable different levels of access (eg, open some doors but not others). A

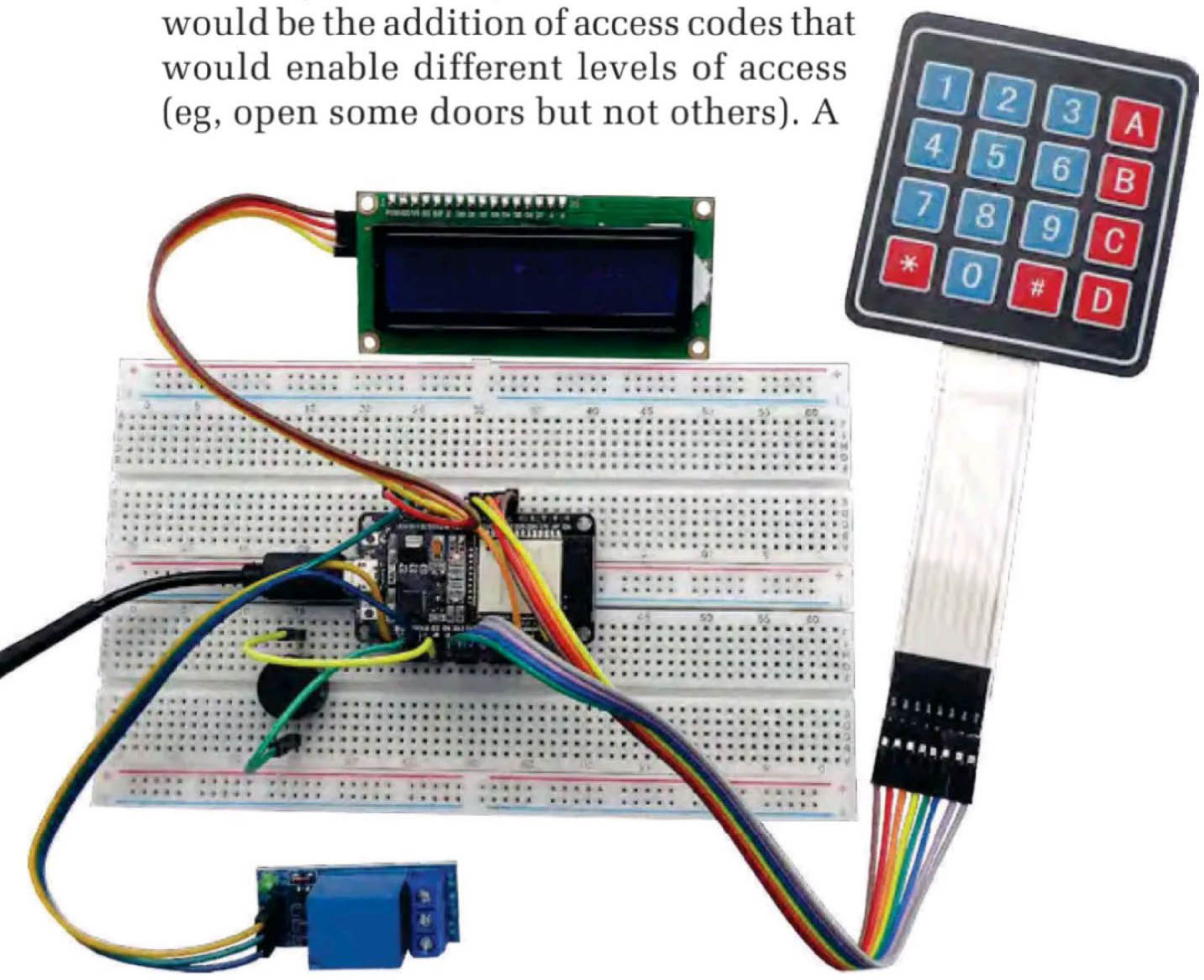


Fig.9.12: The complete ESP32 entry system, showing the backlit 16 × 2 LCD above the breadboard, the 4 × 4 matrix keypad at upper right and the relay at the bottom.

Listing 9.2 – Code for the simple ESP32 entry system

```
// Include the library code:
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>
#include <Password.h>

// Define the LCD display parameters
int lcdColumns = 16;
int lcdRows = 2;
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);

const byte ROWS = 4;  // The keypad has four rows
const byte COLS = 4;  // and four columns
char keys[ROWS][COLS] = {
  { '1', '2', '3', 'A' },
  { '4', '5', '6', 'B' },
  { '7', '8', '9', 'C' },
  { '*', '0', '#', 'D' },
};  // Use an array to hold the key values

byte colPins[ROWS] = { 16, 17, 18, 19 };  // Keypad
byte rowPins[COLS] = { 32, 33, 25, 26 };  // pins
int outPiezo = 2;     // Audible transducer
int outRelay = 15;    // Door actuator relay

Keypad keypad = Keypad(makeKeymap(keys), rowPins,
                       colPins, ROWS, COLS);

Password password = Password(“ABCD”);  // Can change

void setup() {
  pinMode(outPiezo, OUTPUT);
  pinMode(outRelay, OUTPUT);
  // Initialize the LCD display
  lcd.init();
  // Turn on the LCD backlight
  lcd.backlight();
  // Deactivate the door relay
  digitalWrite(outRelay, HIGH); // High to deactivate
  // Set up the number of columns and rows on the LCD
  lcd.begin(16, 2);
  // Print a message to the LCD
  lcd.print(“Enter Passcode”);  // Prompt the user
  keypad.addEventListener(keypadEvent);
  keypad.setDebounceTime(250);
}

void loop() {
  keypad.getKey();
}

void keypadEvent(KeypadEvent eKey) {
  switch (keypad.getState()) {
    case PRESSED:
      lcd.setCursor(0, 1);
      lcd.print(“*”);
      digitalWrite(outPiezo, HIGH);  // Keypress beep
      delay(50);
      digitalWrite(outPiezo, LOW);
      delay(50);
      lcd.setCursor(0, 1);
      lcd.print(“ “);
      switch (eKey) {
        case '#': checkPasscode(); break;  // is right?
        default:
          password.append(eKey);
      }
  }
}

void checkPasscode() {
  if (password.evaluate()) {
    lcd.setCursor(0, 1);
    lcd.print(“* Welcome *”);     //
    digitalWrite(outRelay, LOW);  // Activate relay
    delay(2000);                  // and wait 2s
    digitalWrite(outRelay, HIGH); // Deactivate relay
    delay(2000);                  // afterwards
    lcd.setCursor(0, 1);
    lcd.print(“                “);
   password.reset(); // Reset passcode after valid entry
  } else {
    lcd.setCursor(0, 1);
    lcd.print(“* Invalid *”);
    delay(2000);
    lcd.setCursor(0, 1);
    lcd.print(“                “);
    password.reset();  // Reset after invalid entry
  }
}
```

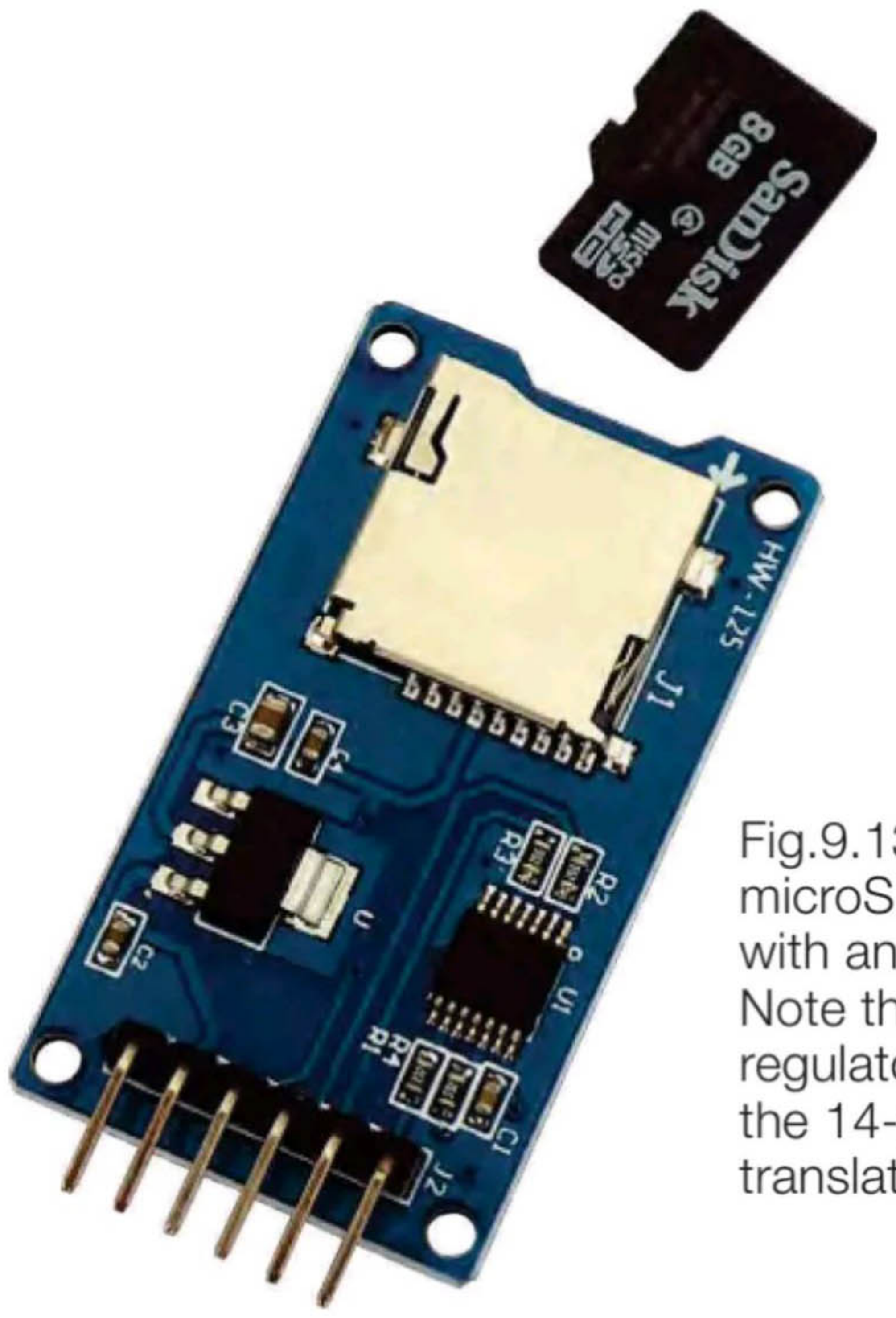


Fig.9.13: A typical microSD card module with an SPI interface. Note the onboard 3.3V regulator (on the left) and the 14-pin logic level translator IC to its right.

lock-out feature could be incorporated to prevent multiple attempts at code entry. This could enforce an obligatory time delay after, say, the third or fourth attempt.

Alternatively, the system could be placed in a 'lock down' mode, triggering an alarm or sending a message to a remote supervisory system.

Coding workshop

For some applications, it can be useful to store data in the form of a file that can be saved and later recalled or transferred to another system for further processing and analysis. This can be accomplished with the aid of an ordinary SD (or microSD) card and some simple code to handle the necessary file processing.

The main difference between an SD card and a microSD card is the size; a microSD card can be placed in a passive adaptor for use in an SD card slot.

The most convenient way of attaching an SD card is using an outboard SD card module with an SPI bus interface like that shown in Fig.9.13.

We previously described the Serial Peripheral Interface (SPI) bus in Part 4 of Teach-in. The bus can operate at high speed over short distances (faster than the I^2C bus) and is thus ideal for transferring data to and from an SD card.

Using the SD Library

The Arduino's built-in SD Library provides you with a useful set of tools for reading from and writing to an SD card. The library supports the popular FAT16 and FAT32 file systems commonly used with standard SD and SDHC cards. Filenames should follow the old DOS '8 + 3' convention, and can include paths separated by forward-slashes, '/'.

It is worth noting that, because the working directory is always the root of the SD card, a name refers to the same file whether it includes a leading slash or not. Thus "mydata.dat" is the same as "/mydata.dat". Note also that the SD library currently supports multiple file opening.

To communicate with an SD card, you will need to make use of the SPI bus available on digital I/O pins 18, 19, and 23. A further pin must be used to select the SD card via the chip select ($\overline{\text{CS}}$) pin. We've used pin 5, but an alternative pin can be specified in the call to SD.begin().

At the start of your code, you will need to include the SPI and SD card libraries:

```
// Include the SD and SPI libraries
#include <SPI.h>
#include <SD.h>
```

Next, you will need to set up the variables that you will be using with the SD library:

```
// Set up variables for use with the SD library
Sd2Card card;
SdVolume volume;
SdFile root;
```

The next step is rather important because you will need to specify the pin number that will be used for the chip select ($\overline{\text{CS}}$) signal to the SD module. Because this varies with SD card modules and shields from different manufacturers, you will need to check the documentation supplied. Here's a line of code that sets I/O pin 5 as the chip select line:

```
// CS signal (change as required)
const int chipSelect = 5;
```

Reading and writing data

Data can be very easily written to an SD card or read from it. First you will need to initialize the card. This can be done using a few lines of code:

```
Serial.print("Initializing SD card...");
if (!SD.begin(10)) {
  Serial.println("Card not ready!");
  return;
}
Serial.println("Card ready!");
```

Notice in this code that we've explicitly stated the pin used for the $\overline{\text{CS}}$ signal (in this case, pin 10). We could do this in a more elegant way by defining the chip select pin as an integer at the start of the code, like this:

```
const int chipSelect = 10; // CS on pin 10
```

We would then need to change the initializing code so that it reads as follows:

```
Serial.print("Initializing SD card...");
if (!SD.begin(chipSelect)) {
  Serial.println("Card not ready!");
  return;
}
Serial.println("Card ready!");
```

Before you can write to a file you need to open it using the SD.open() function, as follows:

```
myFile = SD.open("data.dat", FILE_WRITE);
```

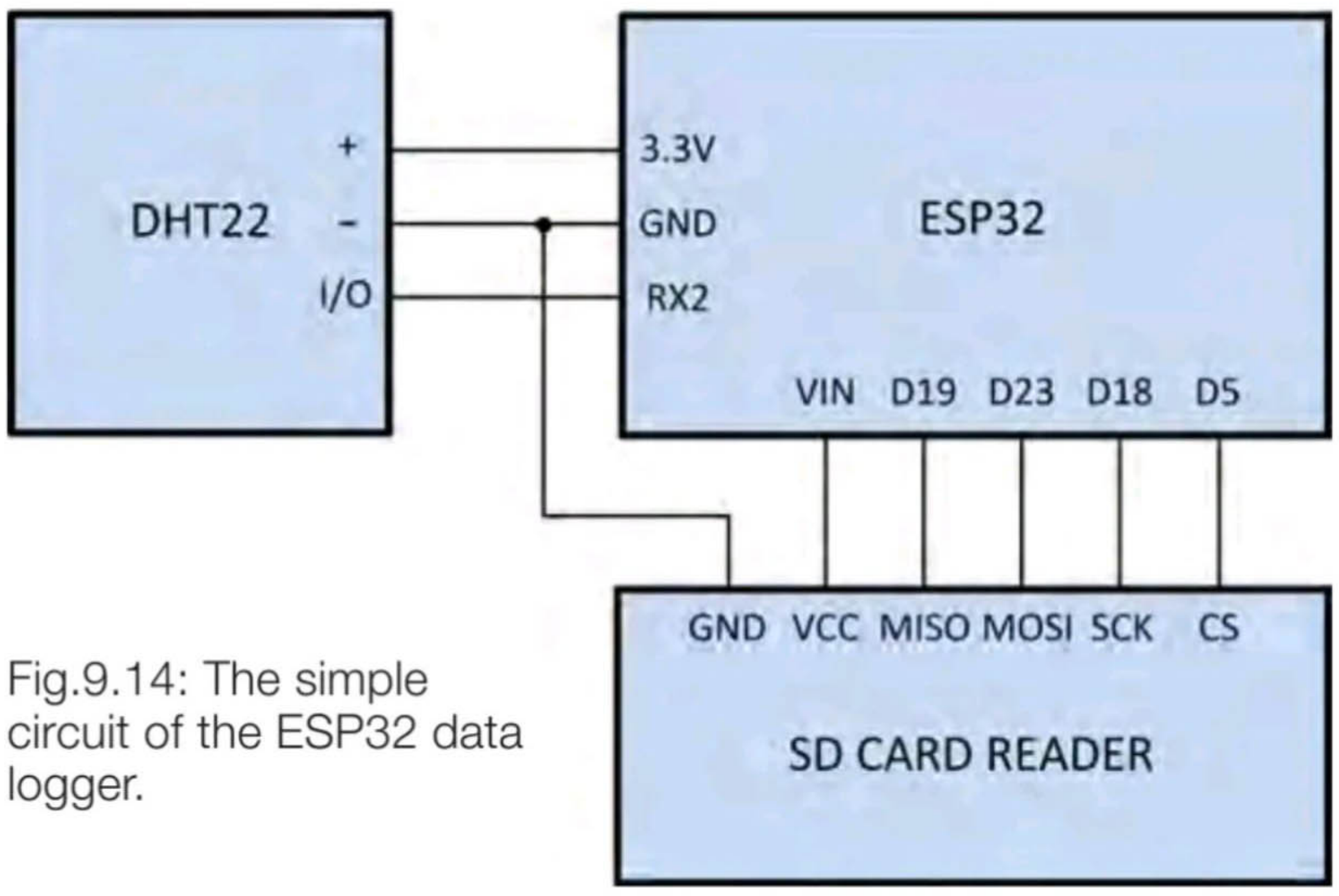


Fig.9.14: The simple circuit of the ESP32 data logger.

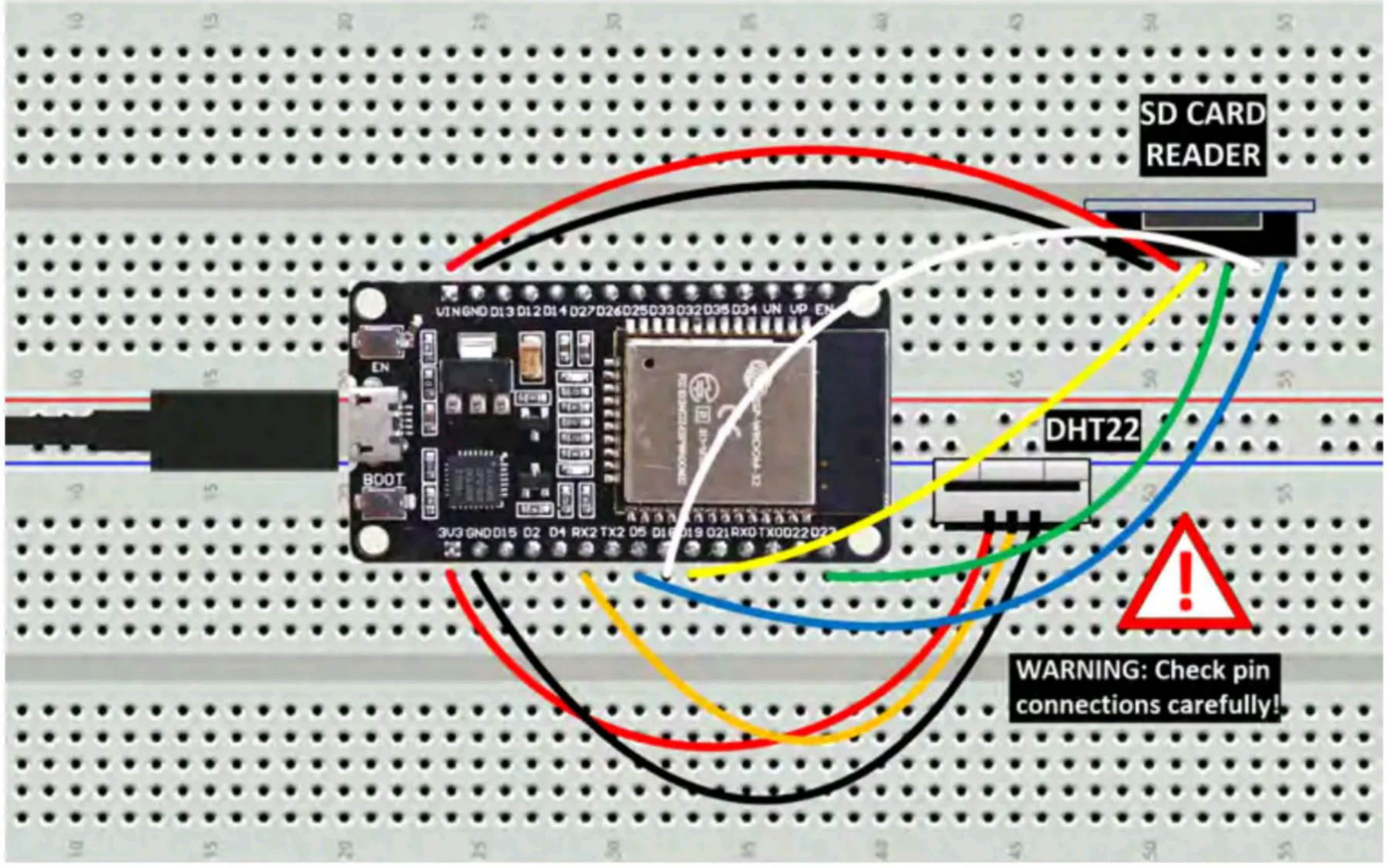

SD CARD READER
DHT22
WARNING: Check pin connections carefully!
VIN GND D13 D12 D14 D27 D26 D25 D33 D32 D35 D34 VN VP EN
3V3 GND D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23
EN
BOOT

Fig.9.15: The breadboard layout and wiring for the ESP32-based data logger.

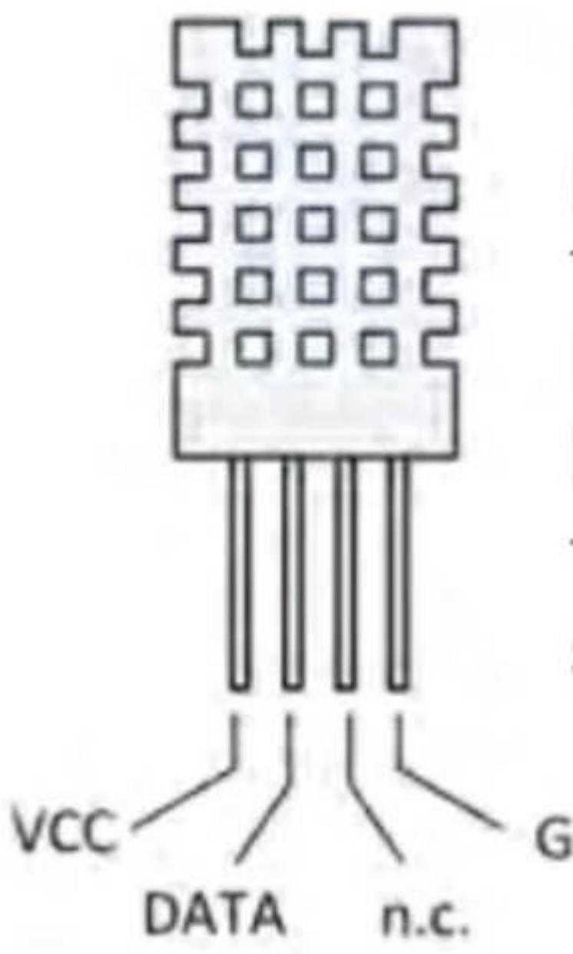


Fig.9.17(a): The pin connections for the **unmounted** DHT22 combined humidity/ temperature sensor.

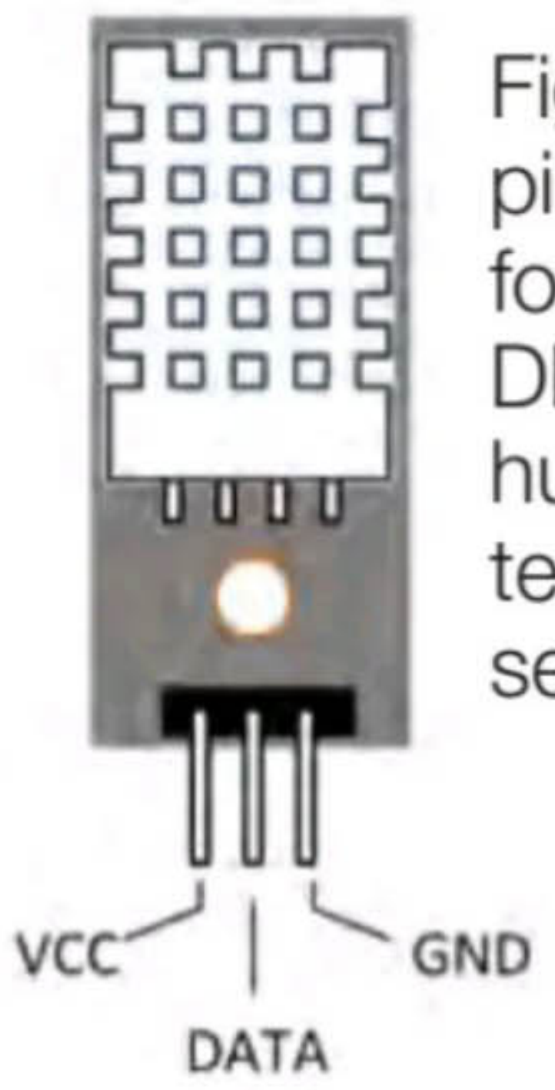


Fig.9.17(b): The pin connections for the **mounted** DHT22 combined humidity/ temperature sensor.

If the file opens without an error, you will be able to write to it. If myFile returns true, the file has been opened, but if myFile returns false, an error has occurred and you will be unable to write to the file.

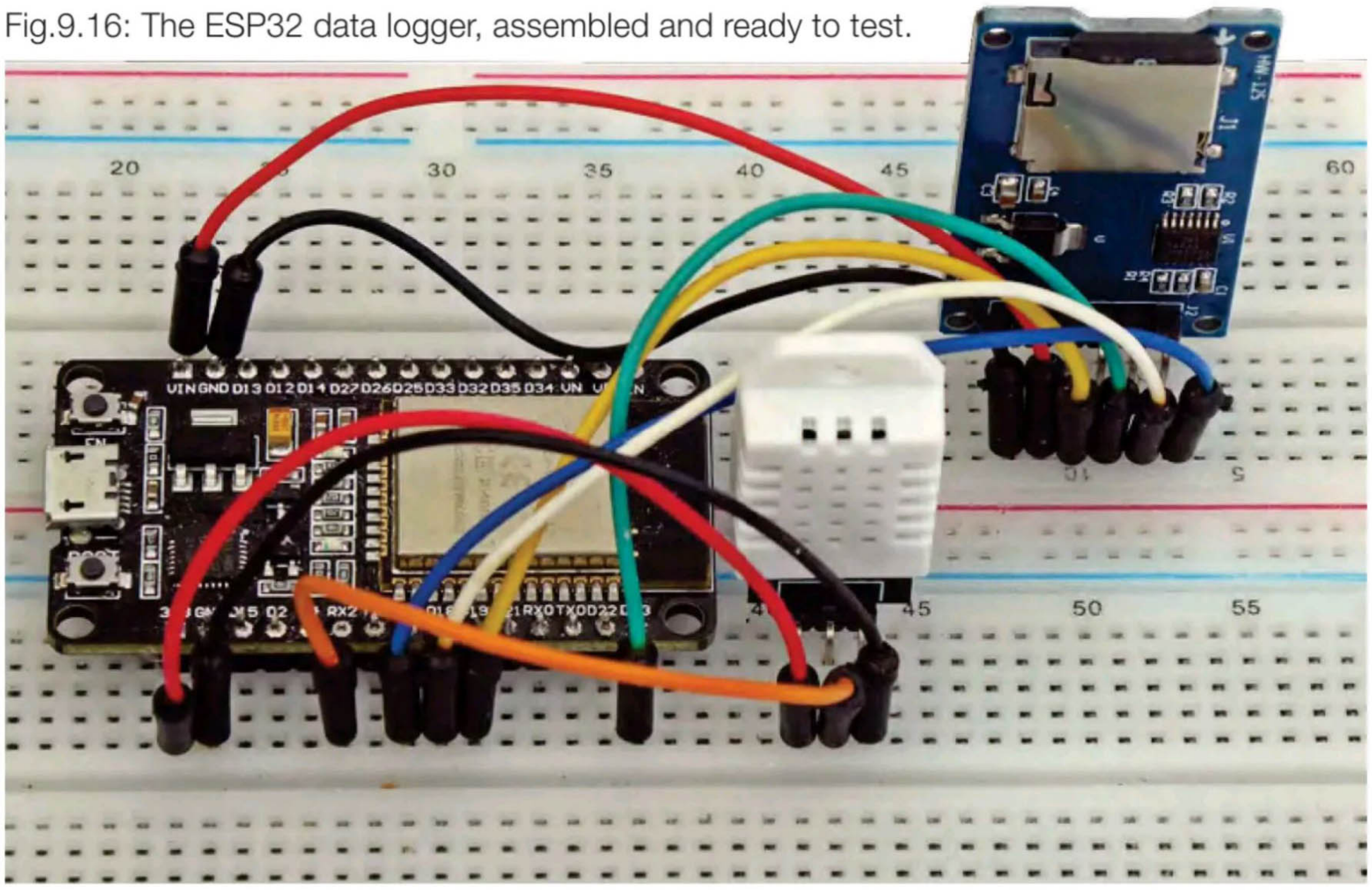
Here's some code that writes a series of values to the file that we've (hopefully) just opened.

Note that you must close the file when you have finished writing to it to avoid data loss; closing the file ensures that any data which has been buffered in RAM is written to the SD card's flash memory:

```
if (myFile) {
  Serial.print("Writing file ...");
  myFile.println("207, 188, 219, 155, 0");
  myFile.close(); // Close the file
  Serial.println(" done.");
} else {
  // It didn't open so print an error message
  Serial.println("Error opening the file!");
}
```

Reading a file is just as easy. First, you need to open the file for reading:

Fig.9.16: The ESP32 data logger, assembled and ready to test.



```
myFile = SD.open("test.txt");
```

Then you need to read data from the file, sending it to the Serial Monitor until you reach the end:

```
if (myFile) {
  Serial.println("test.txt:");
  while (myFile.available()) {
    Serial.write(myFile.read());
  }
  myFile.close(); // Close the file
} else {
  // It didn't open so print an error message
  Serial.println("Error opening the file!");
}
```

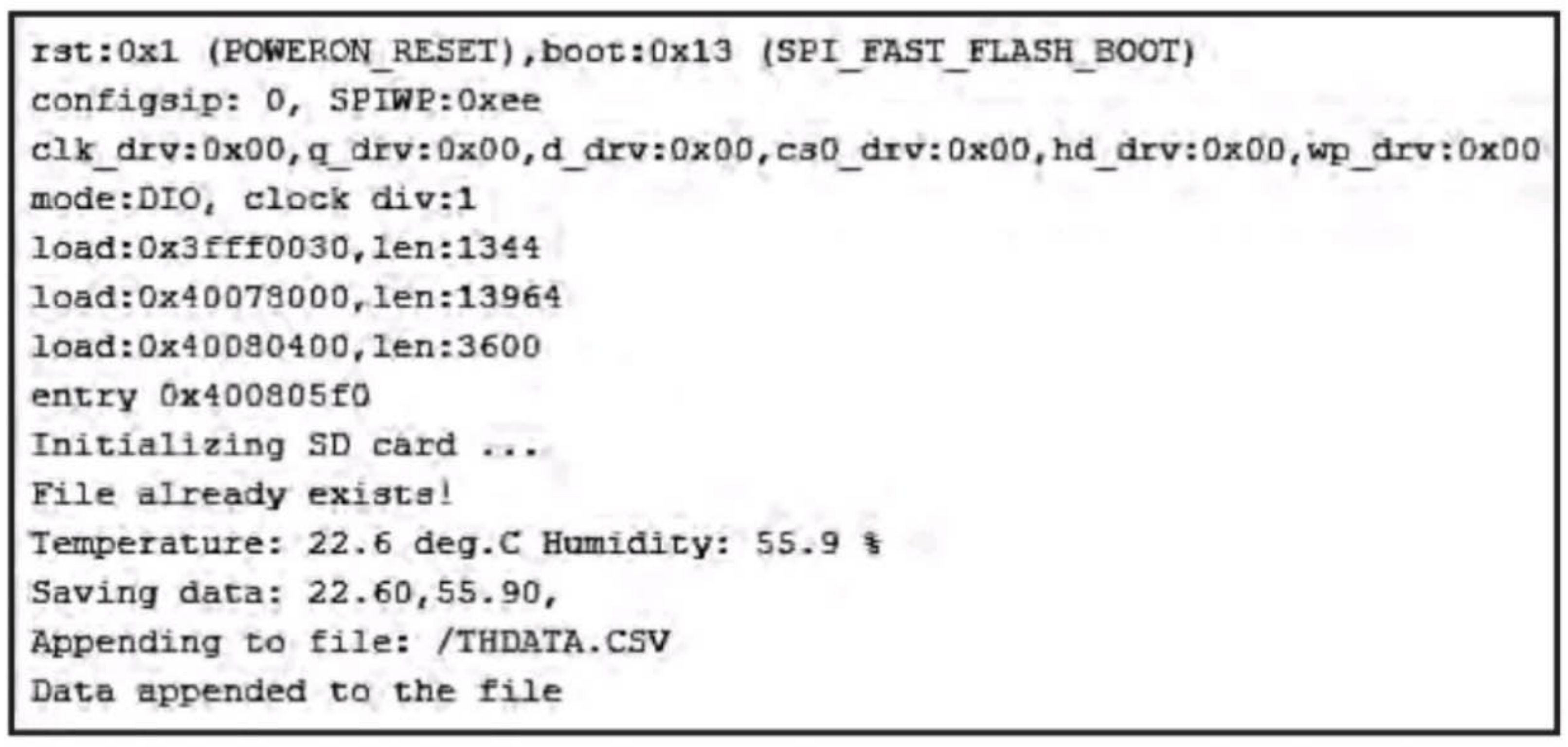
An ESP32 data logger

To put Coding workshop into context, let's look at a simple ESP32 data logger that stores temperature and humidity data obtained from a DHT22 sensor (described in Part 5). The data will be written to an SD card and can then used for later analysis with virtually any spreadsheet package.

Readings are taken and then stored periodically, allowing you to later plot a graph showing how the temperature and humidity changed over time. Having relative humidity and temperature readings also allows you to calculate absolute humidity values later, in case you are interested in that.

The complete circuit of the data logging application is in Fig.9.14 and the wiring layout is given in Fig.9.15, while Fig.9.16 shows the prototype itself. The white DHT22 sensor to the right of the ESP32 is used to obtain temperature and humidity data.

Fig.9.17(a) shows the pin connections for an unmounted sensor, while Fig.9.17(b) shows the pin numbering for a mounted sensor. Since there is some variation in the pin numbering



```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1344
load:0x40078000,len:13964
load:0x40080400,len:3600
entry 0x400805f0
Initializing SD card ...
File already exists!
Temperature: 22.6 deg.C Humidity: 55.9 %
Saving data: 22.60,55.90,
Appending to file: /THDATA.CSV
Data appended to the file
```

Fig.9.18: Serial Monitor diagnostic messages confirming that an SD card is present and the data file has been found.

Gotcha!

It's possible to substitute the lower-cost DHT11 for the DHT22 sensor. However, the 8-bit resolution of the DHT11 humidity/temperature sensor means that the readings will not have a decimal point (the fractional part will be zero). The more expensive (and more accurate) DHT22 fills the fractional parts with data. However, that isn't crucial as the accuracy of the sensor is typically quoted as ±4% for relative humidity and ±2°C for temperature, making the data after the decimal point suspect.

between different manufacturers, it's important to check your module's pin connections before wiring it up (see the Gotcha! panel below).

Listing 9.3 (opposite) shows the complete code for the ESP32 data logger. As always, we've included numerous comments in the code to help you understand what's going on.

The code requires three libraries: DHT.h (or dhtnew.h), SPI.h and SD.h. The first of these provide functions that facilitate the use of the DHT family of humidity/temperature sensors, whilst the second deals with the serial peripheral interface (SPI) used by the sensor, and the third provides the functions associated with reading from and writing to the SD card.

Note that if you are using a very early version of the IDE, you might also need to include the mathematics library that provides the function for rounding the temperature values before they are stored on the SD card, <math.h>. The rounding function is important to remove the fractional part of the values returned from the sensor that might otherwise confuse a conventional spreadsheet program (which might regard the data as text rather than numerical).

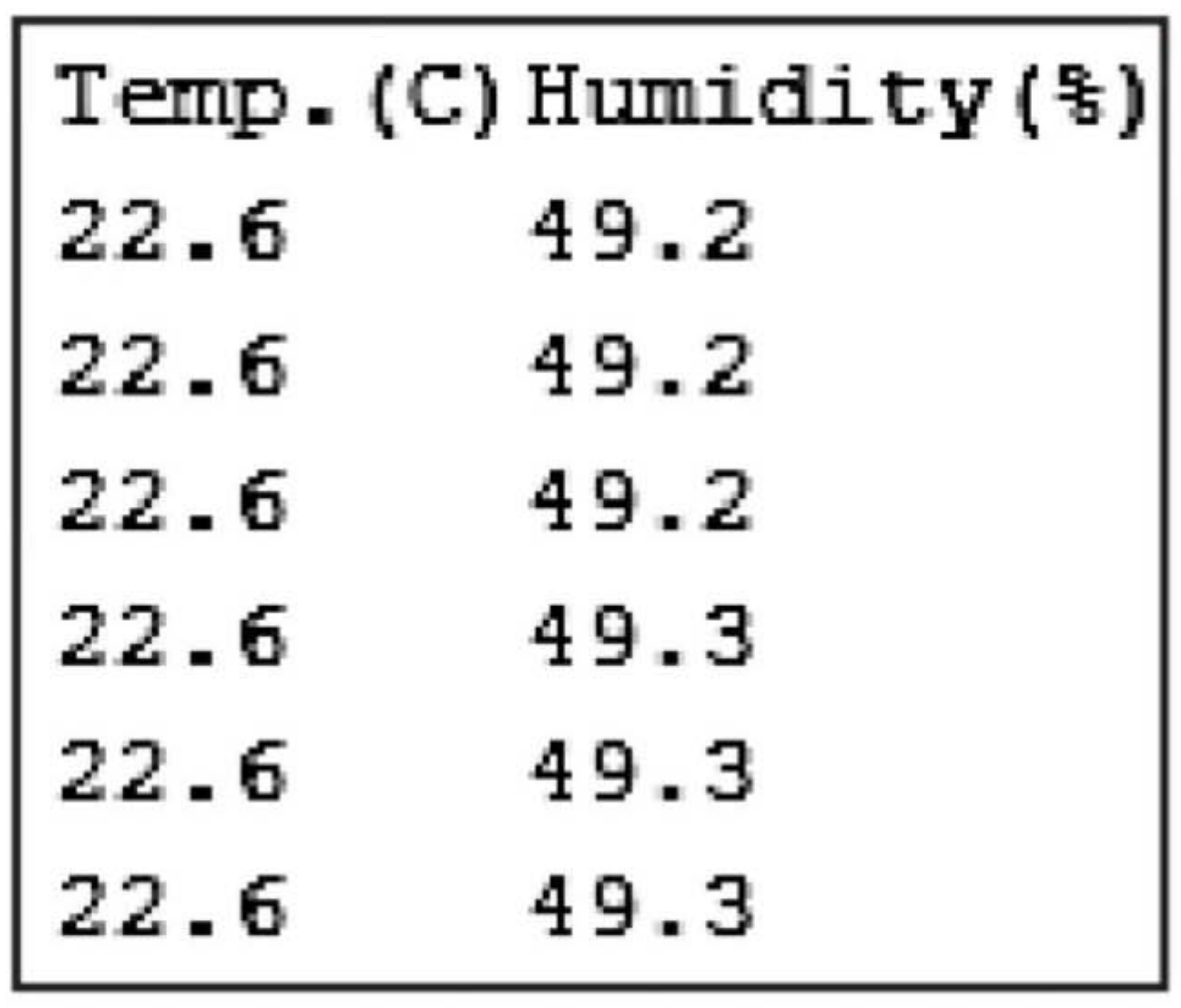
When you've corrected any coding errors that the compiler reports, you will be ready to execute the code. After the ESP32 performs a reset, if all is well you will be greeted by a diagnostic message like that shown in Fig.9.18. If the SD card is not present, or if the SD card interface is not functioning correctly, the program will stop, and an error message will appear. If that is the case, the first thing to check is that you have set the chip select line correctly.

Once the interface is ready, it will start to collect data, writing new information to the SD card on each pass through the loop. You should always wait until the program terminates before attempting to remove the SD card to transfer the data to a PC or other device.

If you run the program several times (without removing the SD card), you will find that further data will be appended to the file (ie, added at the end). If necessary, you can always remove the SD card, place it in a card reader and simply rename the file manually if you want it to start a new file.

The data stored on the SD card is in comma-delimited (CSV) format. On each pass, the data values will be displayed via the Serial Monitor and stored in the file saved to the SD card.

The data displayed by the Serial Monitor should look like that shown in Fig.9.19. If you need to process or visualise this data, you can easily import it into a spreadsheet with each set of values appearing in a different column.



Temp.(C)	Humidity(%)
22.6	49.2
22.6	49.2
22.6	49.2
22.6	49.3
22.6	49.3
22.6	49.3

Fig.9.19: The format of the data stored on the SD card (with the CSV format converted to columns in a spreadsheet).

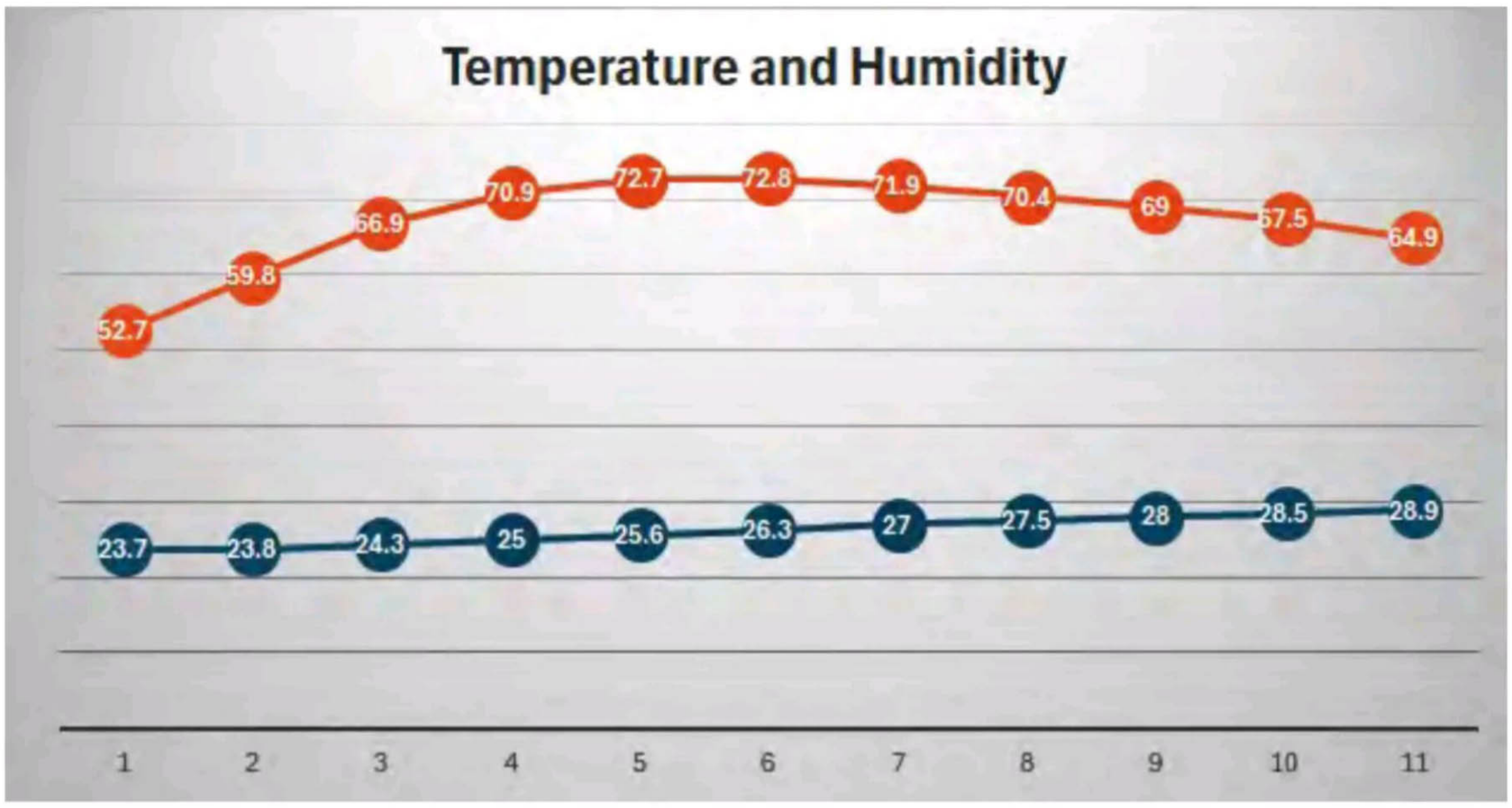


Fig.9.20: Similar data plotted in a chart using Microsoft Excel.

Gotcha!

Some DHT22 manufacturers use different pinouts, so check the pin markings on the module that you are planning to use. If you are using a basic sensor in its four-pin package, connect a 10kΩ resistor between the data terminal (OUT) and the positive supply (VS or VCC). This 'pull-up' resistor is not needed if your sensor is already mounted on a PCB with an onboard pull-up resistor.

Gotcha!

Conventional SD cards are designed for operation from a 3.3V supply and thus it is necessary to ensure that the I/O levels and supply voltage are level-shifted within the hardware interface. This is normally the case with external modules, but it is important to check when modules may have been designed for operation with other systems and may have links or switches fitted.

Listing 9.3 – A simple data logging application

```
#include <dhtnew.h>        // DHT library
#include <SPI.h>           // SPI library
#include "SD.h"            // SD library
#include "FS.h"            // File handling

DHTNEW mySensor(16);       // UART RXD input
#define SD_CS 5            // CS output pin
const int interval = 5000; // Interval between readings
String dataMessage;
String dataString = "";    // A string to hold the data

void setup() {
  Serial.begin(115200);

  // Initialize the SD card
  Serial.println("Initializing SD card ...");
  if (!SD.begin(SD_CS)) {
    Serial.println("SD card not mounted!");
    return;
  }
  // Check the SD card type
  uint8_t cardType = SD.cardType();
  if (cardType == CARD_NONE) {
    Serial.println("SD card missing!");
    return;
  }
  // Open the data file
  File file = SD.open("/THDATA.DAT");
  if (!file) {
    Serial.println("THDATA not found on this card!");
    Serial.println("Creating the file ...");
    writeFile(SD, "/THDATA.DAT",
              "Temperature and humidity data\r\n");
  } else {
    Serial.println("File already exists!");
  }
  file.close();
}

void loop() {
  readTHdata();
  logSDCard();
  delay(interval);  //Wait before writing the next data
}


// Log the current sensor readings
void logSDCard() {
  dataMessage = dataString;
  Serial.print("Saving data: ");
  Serial.println(dataMessage);
  appendFile(SD, "/THDATA.DAT", dataMessage.c_str());
}
```

```
// Write data to the SD card
void writeFile(fs::FS &fs, const char *path,
                           const char *message) {
  Serial.printf("Writing the data file: %s\n", path);
  File file = fs.open(path, FILE_WRITE);
  if (!file) {
    Serial.println("Couldn't open file for writing");
    return;
  }
  if (file.print(message)) {
    Serial.println("File written");
  } else {
    Serial.println("File not written!");
  }
  file.close();
}

// Append data to the SD card
void appendFile(fs::FS &fs, const char *path,
                            const char *message) {
  Serial.printf("Appending to file: %s\n", path);
  File file = fs.open(path, FILE_APPEND);
  if (!file) {
    Serial.println("Couldn't open file for appending!");
    return;
  }
  if (file.println(message)) {  // ln ADDED
    Serial.println("Data appended to the file");
  } else {
    Serial.println("Failed to append the data!");
  }
  file.close();
}

void readTHdata() {
  // Read data from the sensor
  mySensor.read();

  // Then display the returned data
  Serial.print("Temperature: ");
  Serial.print(mySensor.getTemperature(), 1);
  Serial.print(" deg.C\t");
  Serial.print("Humidity: ");
  Serial.print(mySensor.getHumidity(), 1);
  Serial.println(" %\t");

  // Now update the data
  dataString = "";
  dataString += String(mySensor.getTemperature());
  dataString += ",";  // Comma separator
  dataString += String(mySensor.getHumidity());
  dataString += ",";  // Comma separator
}
```

In common with many spreadsheet packages, if you open the data file in Excel, you will need to specify that the file uses ‘delimiters’ and that the delimiter is a comma. You may find that is the default since, while other delimiters are sometimes used, commas are the most common.

Fig.9.20 shows how the data can be represented visually within Excel; other spreadsheet programs, like the free-to-use LibreOffice Calc, can be used to make similar plots. You can also perform numerical analysis, like calculating minimums, maximums, means, standard deviations and all sorts of other useful values. Check your spreadsheet’s help file or manual for details on how to perform those calculations.

As always, there’s a great deal of scope for going further with the ESP32 data logger. Alarms could be set when critical values are detected and outputs could be used to operate indicators, piezoelectric sounders, and relays.

Another refinement might be date/time stamping of the filename so that a unique name is generated every time the program is executed (keeping time was covered last month).

You could also add a column to the data generated that includes the date and time of the measurement, so that you can leave the device running long-term, then plot (for example) a full day’s worth of data by extracting that particular set of data points. The sky is the limit!

Another, related improvement would be to change the main loop so that, rather than simply taking recordings at set intervals, it (for example) makes measurements every minute, on the minute.

Even if you set the interval to be one minute long, it would ‘drift’ over time due to differences in how long each function call took and the exact time spent in the delay function (which can vary slightly between calls). So having a real-time clock or using a timer to determine when to make measurements would give more consistent results.

All of those enhancements would be relatively simple to add and would provide you with a great opportunity to further develop your coding skills and learn more about how the ESP32 works.

Check Point

Now that we've complete our Teach-in series, Check Point replaces our usual Teach-in challenge. This is a great an opportunity to test your knowledge of the ESP32. Just select one answer to each of these questions. When you've finished, you can check your answers at the end of the page. Good luck!

1. To use a library file called WebServer.h in your code, which one of the following lines would you use?
(a) #include WebServer
(b) #include "WebServer.h"
(c) #include <WebServer.h>

2. Multi-line comments are enclosed within:
(a) * and *
(b) // and //
(c) /* and */

3. An IPv4 address requires:
(a) two bytes
(b) four bytes
(c) eight bytes

4. On an I^2C bus, the lines are:
(a) usually pulled high; driven low when data is present
(b) usually pulled low; driven high when data is present
(c) always floating regardless of whether data is present

5. A digital input is fitted with a pull-up resistor. In the absence of any connection, this input will be in:
(a) a low state
(b) a high state
(c) an indeterminate state

6. When using Wi-Fi, an ESP32:
(a) can only be configured as a Server
(b) can only be configured as an Access Point
(c) can be configured either as an Access Point or a Server

7. Errors introduced in the calculations performed by a GPS/GNSS receiver are expressed in terms of:
(a) dilution of precision (DOP)
(b) a percentage of erroneous values (PEV)
(c) received signal strength indication (RSSI)

8. Which serial bus uses SDA and SCL signals?
(a) I^2C
(b) SPI
(c) USB.

9. Why is an LCD panel often fitted with an I^2C add-on?
(a) To improve response time
(b) To reduce the number of I/O lines required
(c) To provide a buffer memory to retain display data.

10. To read the state of a digital I/O pin, you would use:
(a) readInput();
(b) inputState();
(c) digitalRead();

11. DHT11/DHT22 temperature/humidity interface via:
(a) an I^2C interface
(b) an SPI bus interface
(c) the ESP32's serial UART interface

12. The individual segments of a typical seven-segment LED display are conventionally labelled:
(a) a to g and decimal point
(b) 0 to 9 and decimal point
(c) 0 to 7 and decimal point

13. MOSI and MISO are signals present on the:
(a) I^2C bus
(b) SPI bus
(c) UART serial bus.

14. How many keys are present on a 4 × 4 matrix keypad?
(a) 4
(b) 8
(c) 16

15. Internal pull-ups and pull-downs are available for:
(a) all the ESP32's digital I/O pins
(b) none of the ESP32's digital I/O pins
(c) some of the ESP32's digital I/O pins

16. Wi-Fi Protected Access (WPA) uses:
(a) a pre-shared key
(b) a trusted certificate
(c) an encrypted passphrase

17. When x = 1 and y = 0.5, the expression pow(x, y) evaluates to:
(a) 0.5
(b) 1
(c) 2

18. A Hall effect sensor responds to:
(a) electric charge
(b) magnetic field strength
(c) temperature and humidity

19. Which one of the following lines contains an error?
(a) int a == 1.59;
(b) pinMode(32, INPUT);
(c) const byte ROWS = 4;

20. When a function is declared void, it:
(a) does not return a value
(b) returns a value that's discarded
(c) generates an error when the code is executed

Answers:
1. c 2. c 3. b 4. a 5. b 6. c 7. a
8. a 9. b 10. c 11. c 12. a 13. b 14. c
15. c 16. a 17. b 18. b 19. a 20. a

How did you do? If you scored 19 or more, you've become an ESP32 expert! If you scored between 16 and 18, you're nearly there. It's well taking another look at the earlier parts of this Teach-in series if you scored 15 or less.

Robots and ukuleles

Techno Talk

Max the Magnificent

It took 120 years from the first balloon flight (1783) to the first airplane flight (1903). The first artificial Earth satellite came only 54 years later (1957). Another 12 years saw the first human on the moon (1969). Then along came the internet, IoT, AI, VR, AR… hold on tight!

Recently, I was talking to someone much younger than myself. Sad to relate, this age qualifier now applies to most people on the planet. I remember when everyone around me was older; now it's the other way around! It's funny how that works. Even my friends are old (I didn't see that coming).

We have three little girls living next door to us. They believed me when I told them that one of my chores when I was their age was chasing dinosaurs out of our family cabbage patch. That was obviously a joke – my family never even owned a cabbage patch.

All this set me to thinking about how much technology has changed since I was a lad (cue 'travelling back in time' visual and sound effects…).

Magic Robots

I was born in 1957. I bet you're saying, "Wow! That's ages ago!" It is. That's why it has taken me so long to get from then to now.

One way to think about it is that I was born only 12 years after the end of World War II (even back then everything had to have a sequel). My mother was born nine years before the start, and my dad was born in the middle of World War I, also known as the Great War, although I can't figure out what was so great about it.

Kids' toys these days would have been the stuff of dreams when I was a nipper. When I was about eight years old, my parents gave me a toy called The Amazing Magic Robot. The game was presented in a large flat rectangular box, with the cover showing an iconic image of three clean, shiny, well-dressed, beaming kids staring at the robot.

When you removed the lid, you found the 'robot' mounted in the center. Itwas plastic, around three inches (75mm) tall, and holding a pointer. A paper sheet spanning the box had a circle of questions surrounding a hole with a small notch on the left. There was a corresponding circle of answers surrounding a smooth area on the right.

The game came with multiple double-sided sheets of questions and answers. You applied the sheet of interest, placed the robot in the hole on the left (a tab on its base complemented the notch in the side of the hole) and you rotated the robot to point at one of the questions.

When you subsequently transferred the robot into the circle on the right, it magically spun around to point to the correct answer all by itself!

Of course, I now know that this was achieved using magnets, but it was as close to magic as you could get back in the day.

A few years ago, a member of the *PE* community completely unexpectedly sent me an original 1960s version of this awesome antique. It arrived out of the blue and I have it prominently presented in my office.

The computerised toys of today are so much more advanced than those of my youth, and the kids who play with them are so much more knowledgeable than my chums and I ever were. Now I'm wondering what the little girls next door would think about my Amazing Magic Robot. I'll have to bring it home to show them and see their reactions.

There's an app for that!

If you look around, you invariably see young people engaged with their smart phones. Many of them are texting each other, but a large number are interfacing with apps of one form or another. Currently, around two million apps are available in Apple's store. That's a lot of apps!

I'm not immune myself. In addition to a handy-dandy compass app, which would be very useful in the woods (if I ever went there) and a clinometer app, which would be very useful when attaching shelves to a wall (if I ever wished to do so). I have apps for all seasons, from helping me to locate my car to generating sequences of random numbers.

Since a modern smartphone is a very powerful portable computer, it makes sense to use it for a wide variety of tasks.

A few years ago, for reasons I no longer recall, I decided to build a ukulele, as one does. Once you have a ukulele in your hands, you need to tune it.

Originally, I was planning on visiting a local music store to see if they sell such a thing as an electronic ukulele tuner (they do), but then I thought, "I wonder if there's an app for that?" As you'll see if you search the app store, the real trick is deciding which of the myriad ukulele tuner apps to use.

Which way is the wind blowing?

When I was a kid, adults were very good at predicting the weather, especially during or after it happened. Whenever I was walking to the shops with my mum and we bumped into people she knew, one of the main topics of conversation was the weather that day, how it compared to the past 50 years and what we might expect for the next decade or so.

In the early 1960s, there were only two channels on television (BBC and ITV) and they were only on for a small part of the day. The weather forecasts in those days were little more than educated guesses.

Things got better over time, but not much. I still remember watching the forecast on the 15th of October 1987, when BBC weatherman Michael Fish told us that a woman had called in saying she'd heard a hurricane was on the way and his message to her was, "don't worry, there isn't". A few hours later, the worst storm in 300 years killed 19 people and flattened 15 million trees.

I currently live in Huntsville, Alabama, USA. We have three local television stations and goodness only knows how many Doppler radars. In the event of a storm, we turn to any of these channels to be presented with a blow-by-blow account (pun possibly intended) of what's happening.

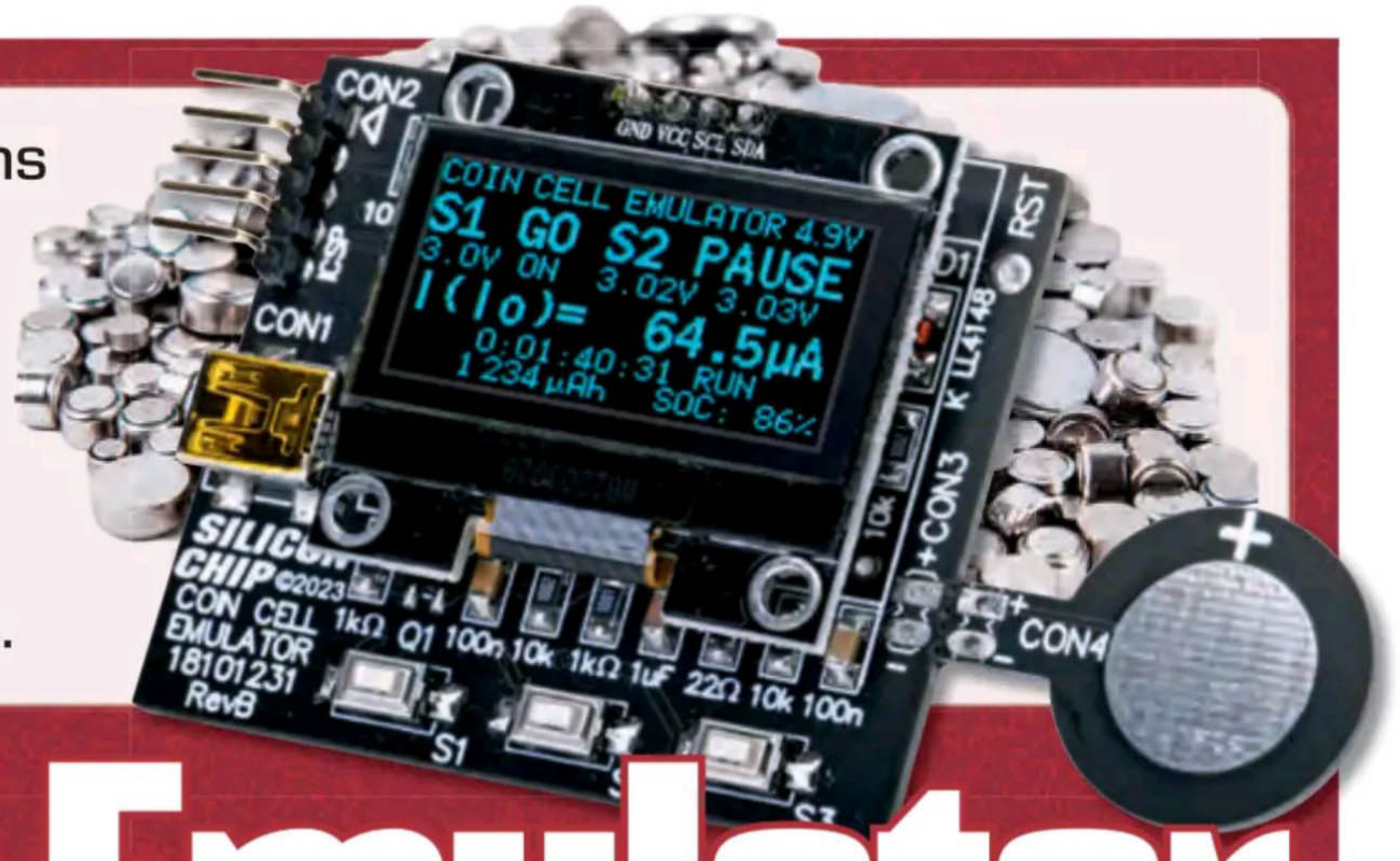
Accompanied by high-resolution imagery and awesome graphics, the presenter literally says things like, "The maximum winds will be crossing County Line Road at 10:21, and you can expect them in Harvest three minutes later at 10:24, then in…"

Hold onto your hat

The pace of technological development is increasing exponentially. The more I think about what's come before, the more I realise I have no clue what is to come. I can't wait to see what technologies we will have when I celebrate my 100th birthday in 33 years!

Project by Tim Blythman

When designing or testing a device that runs from a coin cell, you need to know how much current it draws to determine the cell's life. That can be difficult given the low currents often involved. This device will power such a circuit while showing the voltage, current and other helpful statistics.



Coin Cell Emulator

We have published many designs powered by coin cells (usually the CR2032). They must not draw an excessive current; a high current draw reduces the cell life and causes its voltage to sag due to internal resistance. Coin cells also exhibit a reduced capacity at high discharge rates, compounding the effect.

While many circuits can be characterised with a standard multimeter, that doesn't work well for this type of circuit. A typical multimeter's shunt on the microamp range has quite a high resistance; values around 100Ω are typical.

That is OK for readings in the microamp range, but when the current draw might briefly jump to 5mA or so, the meter is suddenly dropping half a volt, which can change the circuit behaviour substantially. In other words, the burden voltage starts to dominate the reading.

One possible solution is the MicroCurrent DMM Adaptor from the May 2011 issue. That article discusses burden voltage in detail.

However, this Coin Cell Emulator does more than just measure current. It can accumulate the current readings to calculate a capacity value in mAh. It can also produce a varying voltage, so you can test how your circuit behaves as the cell discharges.

The Emulator can also mimic some of the non-ideal characteristics of coin cells, such as internal resistance & voltage fall-off as the battery discharges.

Design

Like the MicroCurrent DMM Adaptor mentioned earlier, the Coin Cell Emulator uses the MAX4238/MAX4239 ultra-low offset, low noise precision op amp to sense very small currents without influencing them.

This op amp has a typical input offset of 0.1μV and an input offset current of 1pA. These are a few orders of magnitude lower than we are trying to measure, so they are unlikely to interfere with our readings.

This is a necessary feature but insufficient to ensure we can measure a wide range of currents. Our design has an upper limit of around 200mA but can measure down to 0.1μA. To do this across a single range would require an ADC (analog-to-digital converter) with 21 bits of resolution.

Instead, our design uses two ranges and a 12-bit ADC that's built into the microcontroller. Oversampling (making multiple measurements and averaging them) gives us a few more bits of resolution, providing the necessary dynamic range.

Circuit details

Fig.1 is the circuit diagram for the Emulator. 5V power comes in via mini-USB connector CON1, with a 10μF capacitor providing board-level supply bypassing. IC1 is an eight-bit PIC16F18146 microcontroller that controls and monitors the Emulator's operation.

IC1's internal DAC (digital-to-analog converter) can deliver 0-4V from pin 17. Unlike some older PICs, the DAC on the PIC16F18146 has an internal buffer and thus has a reasonable drive strength.

The DAC voltage goes to NPN transistor Q1's base via a 1kΩ resistor. Q1 is configured as an emitter follower, so its emitter ranges from 0V to 3.4V, about one diode drop below the 4V maximum from the DAC. Its collector is connected to the 5V rail.

The emitter-follower relies on the reasonably constant base-emitter forward voltage of around 0.6V. Assuming the base voltage is constant, the

Features & Specifications

- Emulates the properties of a coin cell, including internal resistance and discharge over time
- Emulates reduced capacity at high currents
- Adjustable voltage
- Current and charge measurement
- Stopwatch/Timer
- Automatically stops when a threshold voltage is reached
- Dummy PCB can be slotted into a coin cell holder
- Voltage setting: in 0.1V steps
- Typical accuracy: 1%
- Current measurement: 0.1μA resolution up to 200mA
- Charge measurement: 1μAh resolution up to 9Ah
- Voltage measurement: 0.01V resolution up to 3.4V
- Time measurement: 1s resolution up to 999 days

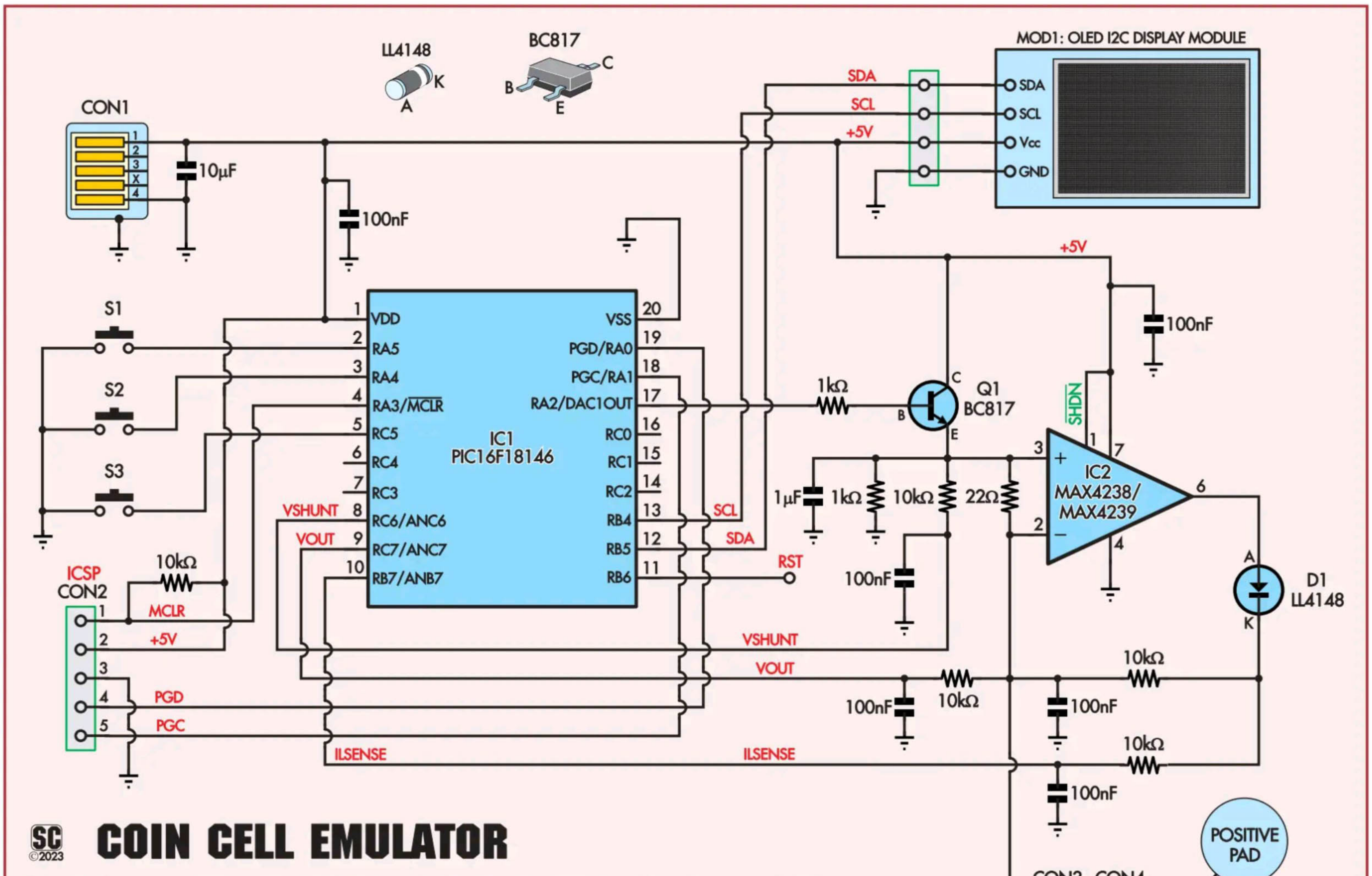

LL4148
K
A
BC817
B
C
E
CON1
1
2
3
X
4
10µF
100nF
MOD1: OLED I2C DISPLAY MODULE
SDA
SCL
Vcc
GND
+5V
S1
S2
S3
IC1
PIC16F18146
VDD
RA5
RA4
RA3/MCLR
RC5
RC4
RC3
RC6/ANC6
RC7/ANC7
RB7/ANB7
VSS
PGD/RA0
PGC/RA1
RA2/DAC1OUT
RC0
RC1
RC2
RB4
RB5
RB6
RST
1kΩ
Q1
BC817
1µF
10kΩ
22Ω
SHDN
IC2
MAX4238/
MAX4239
D1
LL4148
ICSP
CON2
MCLR
PGD
PGC
VSHUNT
VOUT
ILSENSE
CON3
CON4
POSITIVE PAD
NEGATIVE PAD
SC ©2023
COIN CELL EMULATOR

Fig.1: the cunning part of this circuit is the op amp feeding current back into the output through the 10kΩ resistor to cancel out the voltage drop across the 22Ω resistor. This allows the circuit to work with two current measuring resistances of vastly differing values, giving it a very wide current measurement range.

transistor switches on harder if the voltage at the emitter drops, increasing the collector-emitter current and raising the voltage at the emitter.

If the emitter voltage rises, the transistor base current decreases, and less current comes in through the collector. So the circuit maintains the emitter voltage at a steady ~0.6V below the base voltage.

A 1µF capacitor provides some filtering and can provide brief bursts of current to the load. The 1kΩ emitter resistor provides a stable load and ensures that the output voltage decreases if the base voltage decreases.

The 22Ω resistor acts as a current measuring shunt, with two of the microcontroller's ADC pins monitoring the voltage across the shunt via 10kΩ resistors. Each ADC pin also has a 100nF capacitor to ground to present a low impedance to the ADC sampling stage.

The ADC pins are labelled VSHUNT, upstream of the shunt resistor, and VOUT, downstream.

The downstream side of the 22Ω resistor is the positive side of the emulated coin cell, with circuit ground being the negative side. This is available at a pair of 2-pin connectors (CON3 and CON4) and a couple of large pads on a circular part of the PCB. This part of the PCB has a pad on each side and can be slotted into some 2032-sized coin cell holders.

Op amp IC2 has its input pins (pins 2 and 3) connected across the 22Ω shunt, with its output (pin 6) feeding back into the low side of the shunt via diode D1 and a 10kΩ resistor. A third ADC pin of IC1 (pin 10; labelled ILSENSE) monitors the voltage at the diode's cathode via another 10kΩ resistor and 100nF capacitor arrangement.

A 100nF capacitor bypasses IC2's 5V supply (pin 7) and ground (pin 4). Pin 1 ($\overline{\text{SHDN}}$) is also pulled up to the 5V rail, allowing the op amp to operate normally when powered.

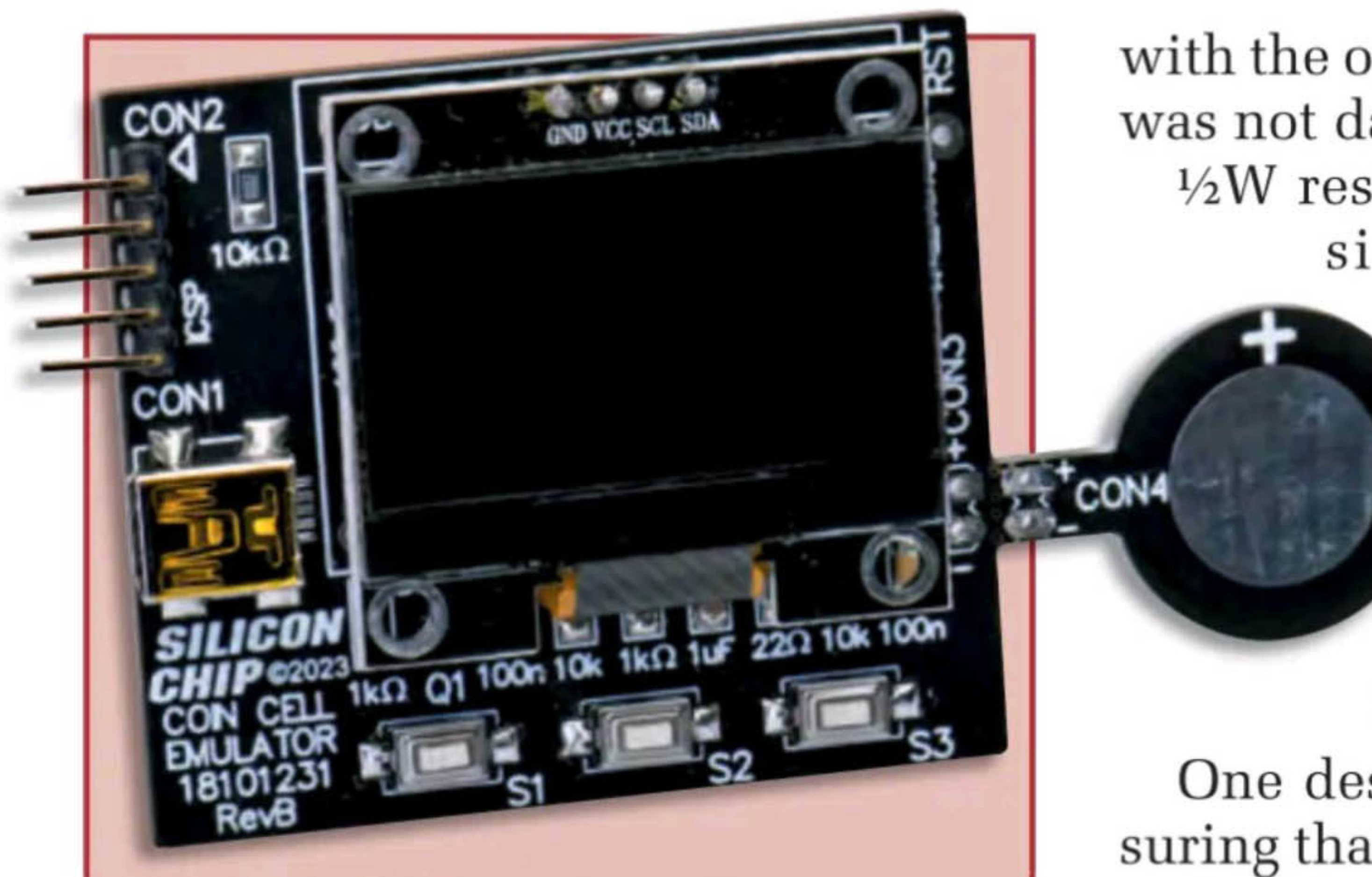
Op amp operation

If a small current flows through the 22Ω resistor, a voltage appears across the op amp's input terminals and its output rises. Current flows through the diode and 10kΩ resistor back to its inverting input and the downstream end of the shunt resistor. The diode ensures the op amp can only source and not sink current.

Effectively, the op amp overrules the shunt and supplies current to the output of the Emulator. Smaller currents can be sensed by measuring the voltage across the 10kΩ resistor and applying Ohm's Law.

Eventually, the op amp output saturates and cannot supply enough current. It has nearly rail-to-rail operation, so its maximum output is around 4.9V. Assuming the Emulator output is at about 3V, there is around 1.3V across the 10kΩ resistor, with the op amp supplying around 130µA.

The voltage across the 22Ω resistor can now develop and is measured by the ADC channels connected across it. We can thus measure across a wide dynamic range since we are effectively



The Coin Cell Emulator is a compact but handy development and testing tool. Even if you don't design circuits for coin cell operation, it's a useful low-voltage PSU with current monitoring.

using two shunts with vastly different resistances. Combining the currents is as simple as adding them.

Using a high-side measuring shunt also means that the ground circuit is uninterrupted and can be shared with any other gear that needs to be attached (programmers, debugging gear or other meters) without affecting current readings. This is handy, especially if you are running everything from a computer.

The test point labelled RST was originally included to allow the Emulator to control a connected circuit by pulsing its reset line low. But since the Emulator can power cycle the circuit, we did not implement this feature.

Instead, a nominal 1Hz clock signal is available at this pin. This can be used to trim IC1's internal timer for accurate timekeeping.

Short circuit handling

Let's examine what happens when a short circuit is applied to the output of the Emulator. With the DAC set to its maximum of 4V, around 140mA flows through the 22Ω resistor.

With a typical transistor β (gain) of around 400, the base current is around 350μA and the 1kΩ resistor on Q1's base drops 0.35V, so the voltage at the emitter falls from 3.4V to around 3V. The transistor thus dissipates around 280mW (2V × 140mA), comfortably within its 500mW rating.

The remaining voltage is across the 22Ω resistor and it dissipates around 400mW. That's a bit on the high side for the typical ¼W rating of an M3216/1206-size SMD part. Our prototype got quite hot around that resistor with the output short-circuited, but it was not damaged.

½W resistors are available in this size, so that's what we're specifying. That allows the Emulator to handle a short circuit on its output indefinitely.

ADC input impedance

One design consideration was ensuring that the ADC sampling did not unduly load the Emulator's output. A load of even 1MΩ to ground would be measurable, as it would draw 3μA at 3V.

Two ADC channels are fed directly from low-impedance sources and unaffected by loads; transistor Q1 and op amp IC2 drive the VSHUNT and ILSENSE lines, respectively. Effectively, they are upstream of their respective shunts.

On the other hand, any load applied to the VOUT line would be indistinguishable from a load at the Emulator output. The ADC input used to sense the VOUT voltage is such a load.

The ADC input consists of a small capacitor, nominally 28pF, which is connected to the ADC pin to sample the voltage. The capacitor is then connected to the internal ADC circuitry (and disconnected from the pin) to perform the conversion.

The 'switched capacitor' model can be used to calculate an equivalent DC resistance. A switched capacitor is simply a capacitor that is switched between two different connections at a known frequency. The resistance of such an arrangement is simply 1/CF, where C is the capacitance in farads and F is the frequency in hertz.

With our 100Hz sampling, this comes out to around 350MΩ, which is more than high enough. Higher sampling rates would reduce this apparent resistance.

Another point to consider is that the ADC capacitor is not discharged between samples, so the load presented by the switched capacitor is not equivalent to a load to ground, but rather as a resistance between the different sampling points. That raises its effective resistance.

The PIC16F18146 has an ADCC (analog to digital converter with computation) module. We previously used some of its advanced features in the Digital Boost Regulator that was published in the December 2023 issue.

The differential ADC inputs make it much easier and more accurate to measure the difference between two voltages, as we are doing here. The sampling time is also programmable, so we have extended it slightly to ensure the sampling capacitor can fully settle at the input voltage.

There is also a DIA (device information area) that holds information such as the measured value of the chip's internal voltage references. This means we can measure voltages against this reference without a separate calibration step.

The DAC mentioned earlier is an 8-bit type with a 4.096V (nominal) voltage reference. It can deliver up to around 4V in 16mV steps and can produce a voltage with 0.1V precision.

The output voltage at VSHUNT and VOUT is thus limited by design to around 3.4V. This works well with circuits using 3.3V microcontrollers that typically have a 3.6V upper limit. The MAX4238 op amp specifies a common mode voltage up to around 3.6V (with a 5V supply) and the op amp inputs stay within that range.

Microcontroller and interface

IC1's pins 2, 3 and 5 connect to switches S1, S2 and S3, respectively, with their other sides grounded. The micro applies an internal weak pullup current to each, so it can detect button presses as level changes on those pins.

An I^{2}C OLED module is connected to IC1's pins 12 and 13 for the SDA and SCL signals. The OLED is powered from 5V; it has an onboard 3.3V regulator with I^{2}C pullups, allowing it to interface with a microcontroller running from 3.3V or 5V.

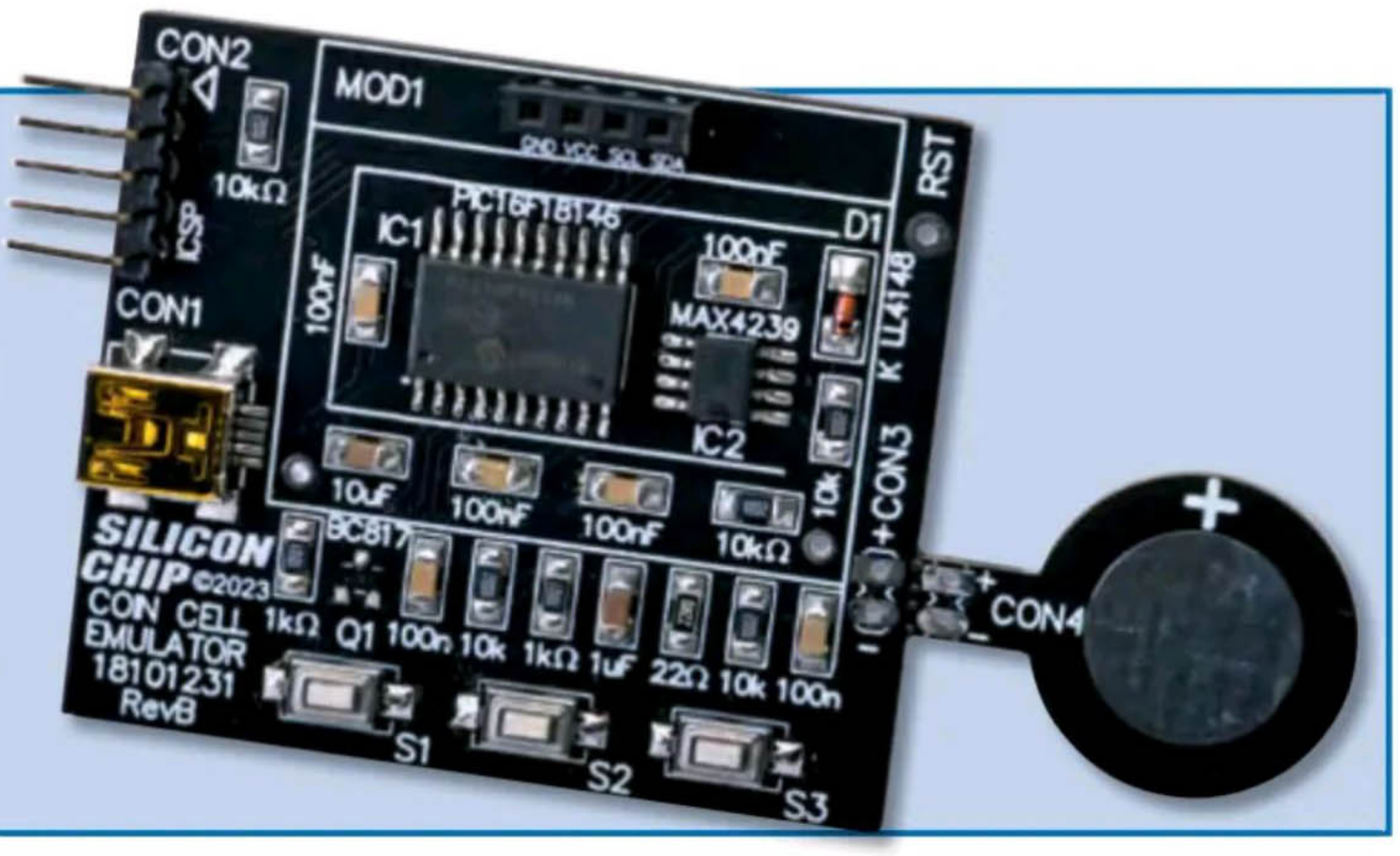
IC1 has a local 100nF bypass capacitor between its pin 1 supply and pin 20 ground. Pin 4 ($\overline{\text{MCLR}}$) is pulled up to 5V by a 10kΩ resistor, allowing the microcontroller to run.

These pins and pins 18 and 19 (PGC and PGD) are taken to CON2 for in-circuit serial programming (ICSP) of the microcontroller.

Coin cell behaviour model

As the saying goes, all models are incorrect, but some are still useful!

There are several characteristics of coin cells that we are explicitly modelling. We're not claiming that the model is comprehensive, but it mimics the behaviour of a real coin cell well enough to be useful.



We have used a socket header to attach the OLED module in our prototype, but the Emulator will be much more robust if you solder the display directly to the main PCB.

Our model is based mainly on a CR2032 cell, as that is what we have used the most. We fitted graphs provided by several CR2032 manufacturers to curves described by simple equations, adjustable by a single parameter.

There is a lot of variation between manufacturers and even between cells from the same manufacturer under different conditions. The default behaviour of the Emulator is similar to a typical coin cell.

Firstly, coin cells have internal resistance. For CR2032 cells, the value is around 20Ω, but it can change with load and state of charge.

Other 20mm diameter cells, such as the CR2016 (half as thick as a CR2032 at 1.6mm), appear to have a similar internal resistance. So the Emulator will also be suitable for thinner cells of the same diameter but might not be as accurate for those with a smaller diameter.

A simple way to model the internal resistance is with a fixed resistor, and we chose the 22Ω part that we have already explained. One advantage of using a fixed resistor is that this resistor can also be used as a current measuring shunt.

The actual circuit appears to have an internal resistance of around 24.5Ω, as the 1kΩ base resistor carries a current in proportion to the load current divided by the β (gain). So it adds around 2.5Ω (1000Ω ÷ 400) of resistance for a β of around 400.

The next factor is that, like most batteries, the terminal voltage drops as the cell discharges until it is flat. For coin cells, the voltage drops a little at the start, then is quite steady for most of the cell life. Once it starts to fall after that, it does so quite dramatically.

While we looked at using a curve to model this, curves that fit all three stages were complex, and we found that they weren't helpful for observing circuit behaviour as the cell goes flat.

Instead, we have implemented a simple model that maintains a flat voltage and then linearly changes the output voltage as the cell's state of charge (SoC) nears its endpoint. For example, with this set to 10%, the voltage is flat from 100% to 10% SoC, then drops to half by 5% SoC. Finally, the voltage is ramped to zero when the Emulator determines the cell is flat.

This feature can be turned off (set to 0%) to disable this behaviour. Fig.2 shows the graph of the data sheet behaviour compared with the emulated behaviour.

While we could have more closely emulated this with, say, four linear sections, we decided not to do that. We found that a constantly changing voltage during use interfered with monitoring the device's operation. In other words, we have sacrificed reality for usability. Our simple voltage curve provides a voltage that behaves very predictably.

It does omit the higher voltage at the start, but that can easily be emulated manually by initially setting the voltage to 3.2V, observing the operation, then manually dropping the voltage to 3V.

Another well-known aspect is that a cell's apparent capacity (in mAh) is reduced if it needs to supply a heavier load. The manufacturers also provide graphs to characterise this behaviour.

One typical graph we saw showed that a nominally 240mAh cell provides only 150mAh with a continuous discharge of 3mA, nearly halving its effective capacity.

We found quite a few curves that demonstrate this behaviour. The data varied quite a bit, but it was clearly some form of polynomial relationship. A good technique for finding the order of polynomial relationships is to take a plot of the logarithms of the variables in question. The order of the polynomial is related to the slope of this graph.

Consider the quadratic equation $y = x^2$. The value of $\log(x^2)$ is equal to $2\log(x)$, for positive values of x, so the graph of $\log(y)$ or $2\log(x)$ against $\log(x)$

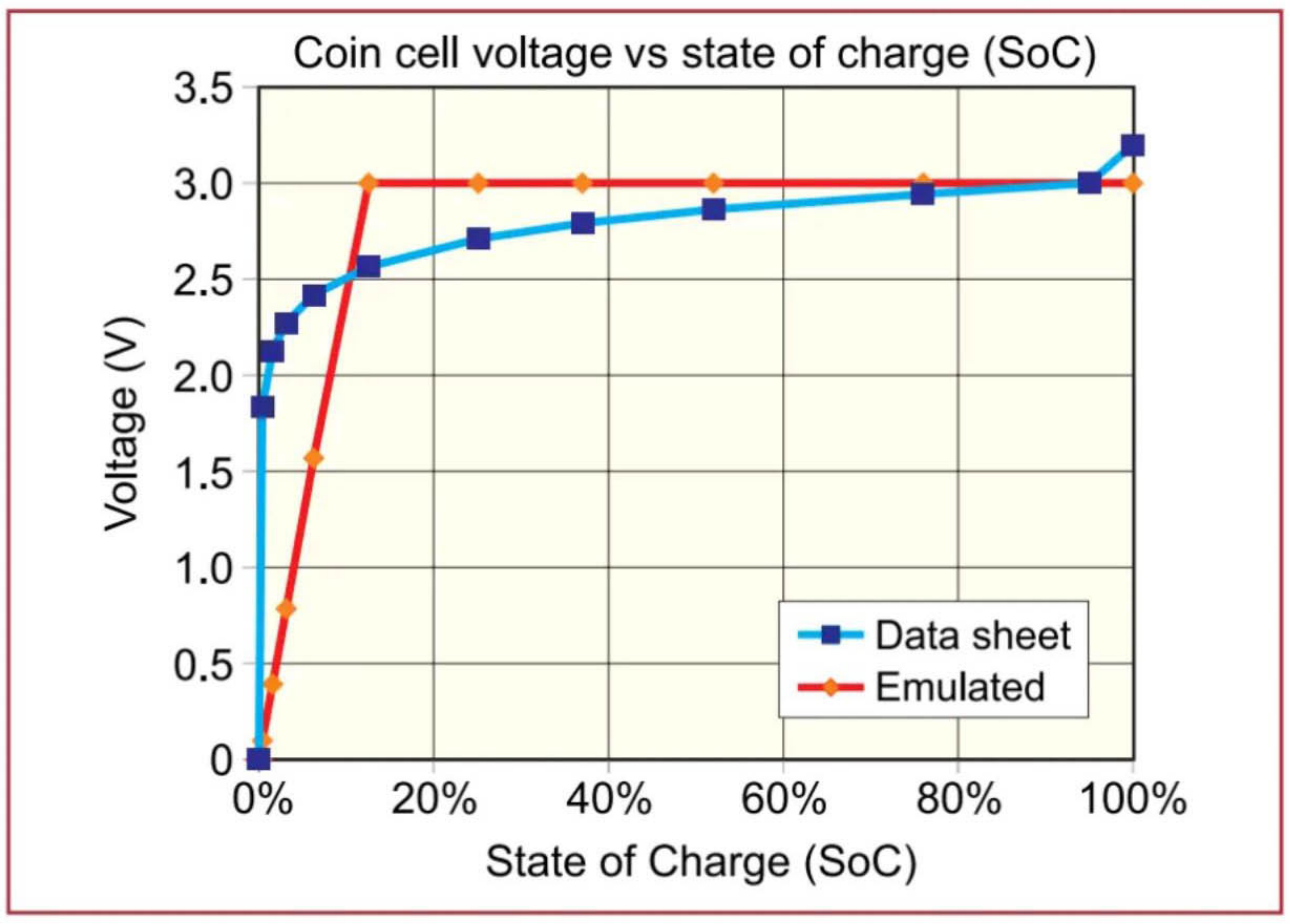


Fig.2: our emulated cell voltage curve is much simpler than that seen in many coin cell data sheets, but it still mimics the cell going flat. Otherwise, we prefer to manually adjust the voltage and observe what happens.

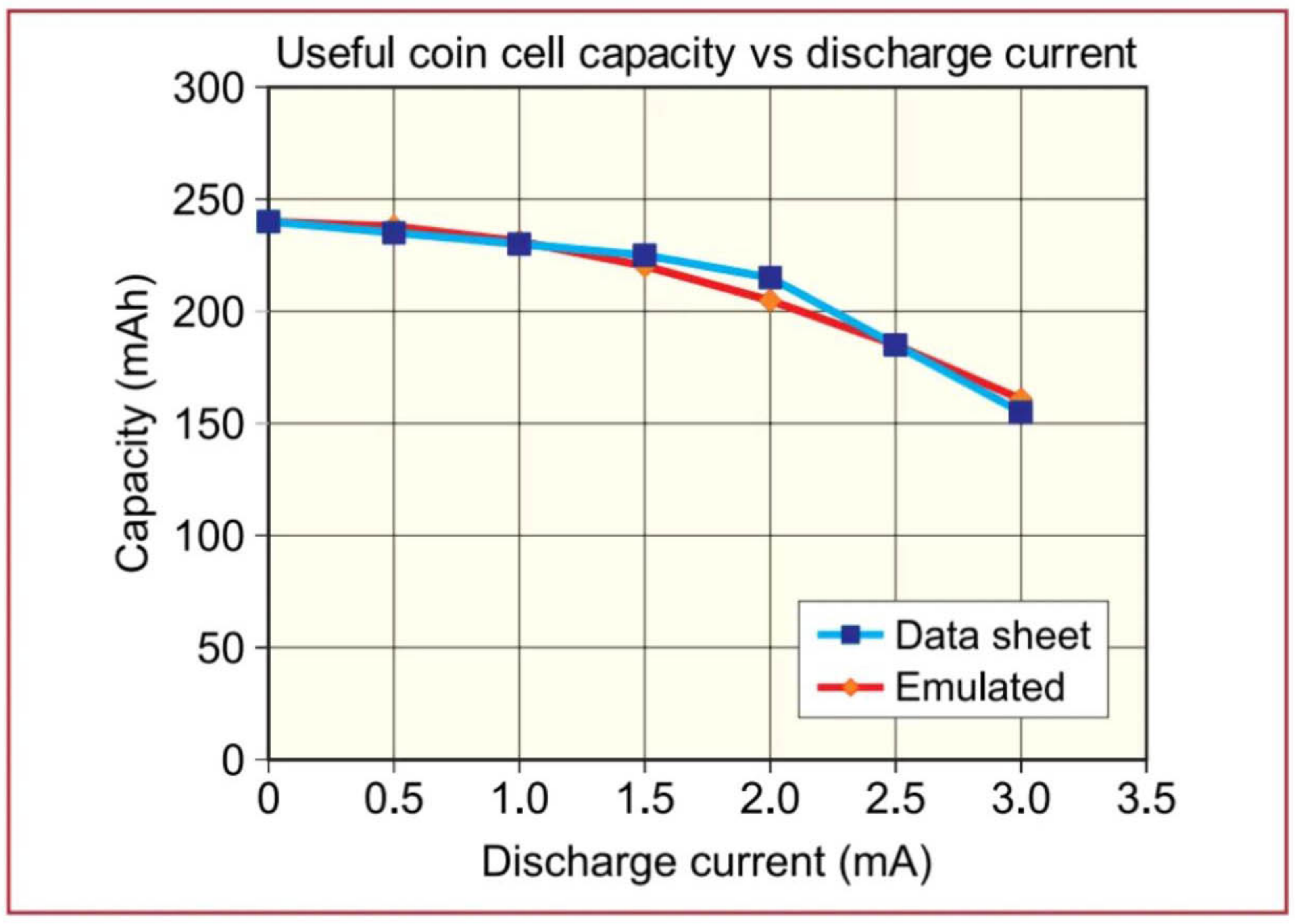
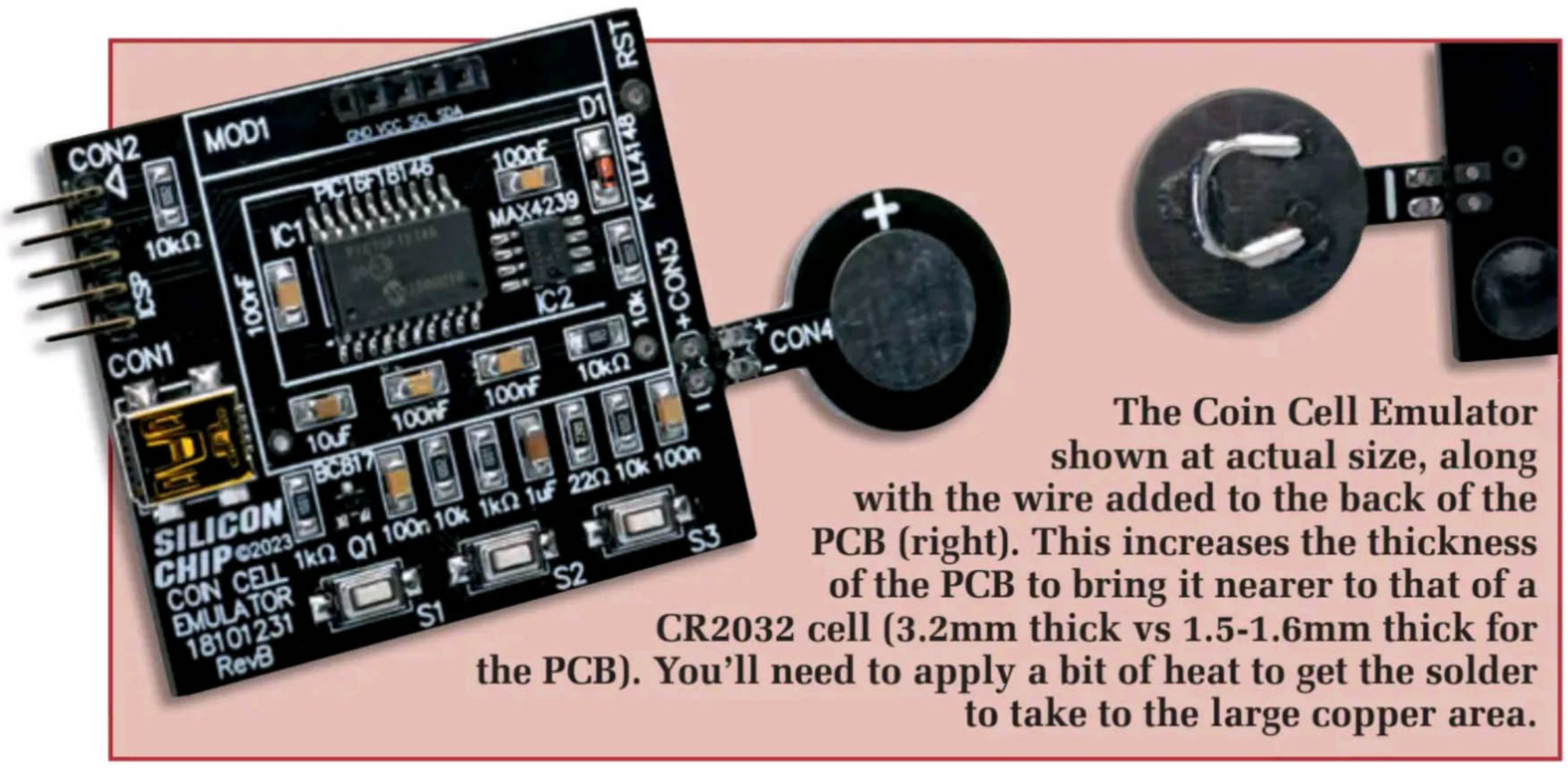


Fig.3: the reduction in useful capacity is modelled as a straightforward quadratic curve. It's a compromise between simplicity and accuracy.



The Coin Cell Emulator shown at actual size, along with the wire added to the back of the PCB (right). This increases the thickness of the PCB to bring it nearer to that of a CR2032 cell (3.2mm thick vs 1.5-1.6mm thick for the PCB). You'll need to apply a bit of heat to get the solder to take to the large copper area.

would have a gradient of two, suggesting a quadratic equation of some sort (a quadratic is a second-order polynomial).

We found that the slopes of these log/log plots were just over two. So we modelled this with a quadratic equation and found that it fit quite well to the manufacturer data and was simple enough for the 8-bit micro to calculate.

We didn't see any charts that show behaviour much above 5mA but this model also allows us to extrapolate. This extrapolation suggests severely degraded capacity as the current enters this region. Our experience is that coin cells discharge very quickly if you draw much more than 5mA from them, so this makes sense.

Our model takes a parameter equal to the current at which the cell capacity is halved. We have used a default value of 3.5mA, which matches the CR2032 data sheets we examined. It also makes it easier to match your Emulator to a specific type of cell if that is required.

If this value is set to zero, then there is no modelling and the Emulator will show the same capacity no matter what current is drawn. Fig.3 shows the graph of the model against typical data from a cell data sheet.

Regarding the short circuit behaviour noted earlier, it should be apparent that, like a real coin cell, the Emulator will quickly 'go flat', effectively ending the short-circuit condition.

Firmware

For the most part, the microcontroller allows the user to set the output voltage, although it can modify that based on the discharge modelling. It monitors the voltages around the circuit and calculates and sums the currents in the two measuring shunts.

A timer keeps track of time intervals and allows the current to be accumulated over time for the charge and capacity calculations. The measured charge (in mAh) is taken from the actual value, while the SoC calculation is based on the modified behaviour at higher currents.

All this information is displayed on the OLED screen. There are modes to allow a test to be started and paused. These tests turn on the output voltage, start the timer and start the charge accumulator. The test can be ended manually or automatically at a previously set endpoint voltage.

Alternatively, the Emulator can simply be used as a power supply that can monitor the current consumed by the circuit under test.

A settings screen can be used to trim the parameters used to set the output voltage. Since the Emulator can measure its output, a calibration routine can set these automatically. You can also trim the resistance values of the shunt resistors and adjust numerous parameters that control the coin cell emulation.

Since the PIC16F18146 has an internal EEPROM memory (which can withstand more write cycles than flash memory), the calibration and setup parameters are immediately stored in EEPROM when modified.

Response time

Figs.4 and 5 show the rise and fall times of the output voltage in response to a change in the setpoint. These charts

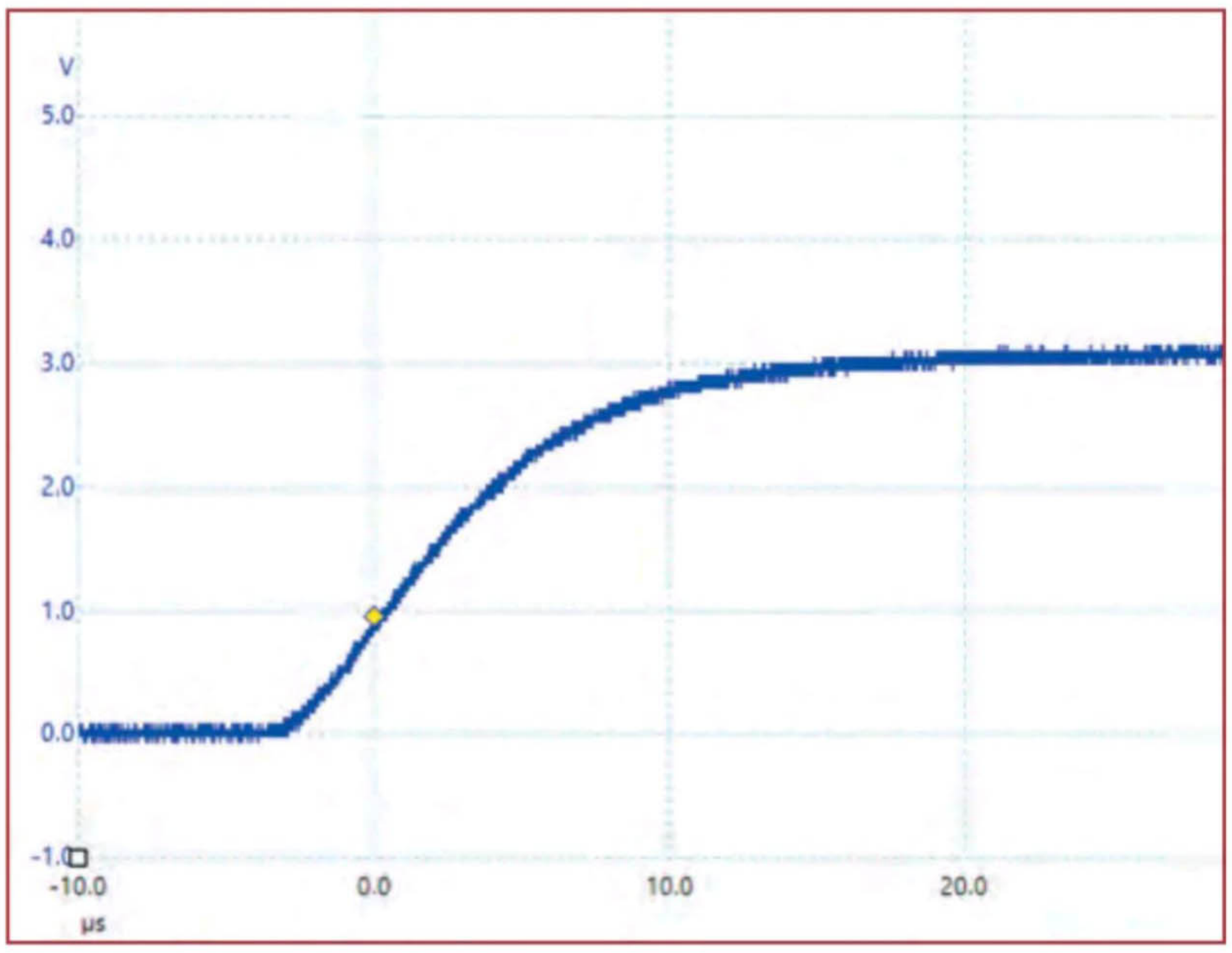


Fig.4: the rise time of the output is limited by the capacity of the circuit to supply the current to charge the 1µF capacitor at its output (the timebase is in µs here). The DAC that controls the voltage has a settling time of around 10µs.

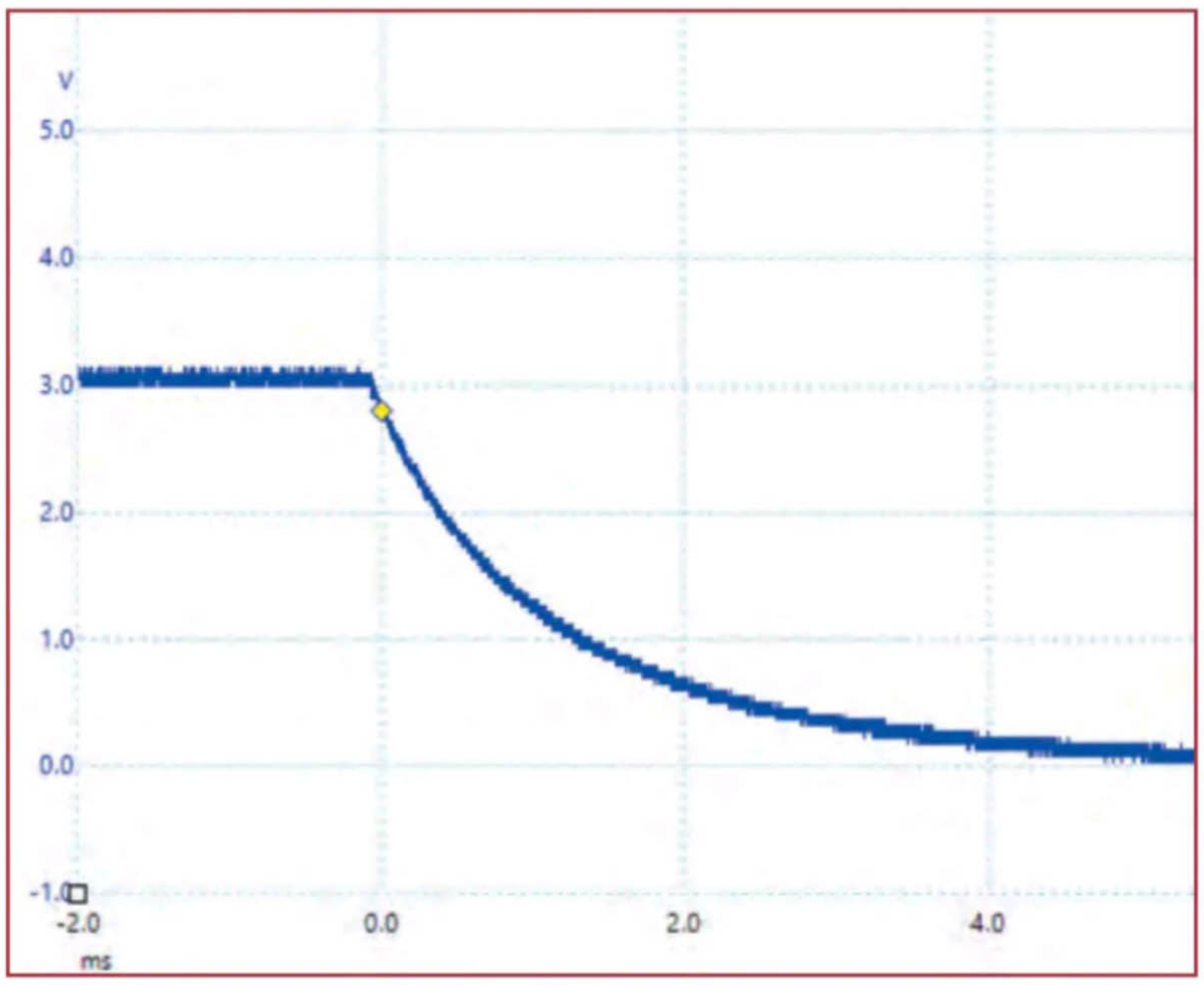


Fig.5: the longer fall time of the Emulator output is almost entirely due to the 1ms time constant of the 1kΩ/1µF RC combination. After about 4ms (four time constants), the voltage settles near its 0V endpoint.

were taken in an unloaded state (although the Emulator accurately indicated the expected 0.3µA draw from the 10MΩ scope probe at 3V!).

As expected, the rise time is short, about 20µs from 0V to 3V. About half of this is due to the 10µs settling time of the DAC, with the other half being the time to charge the 1µF capacitor with the 200mA available.

The fall time is dominated by the 1ms time constant of the 1kΩ/1µF pair and takes about 4ms to settle near its final value. An external load will speed this up.

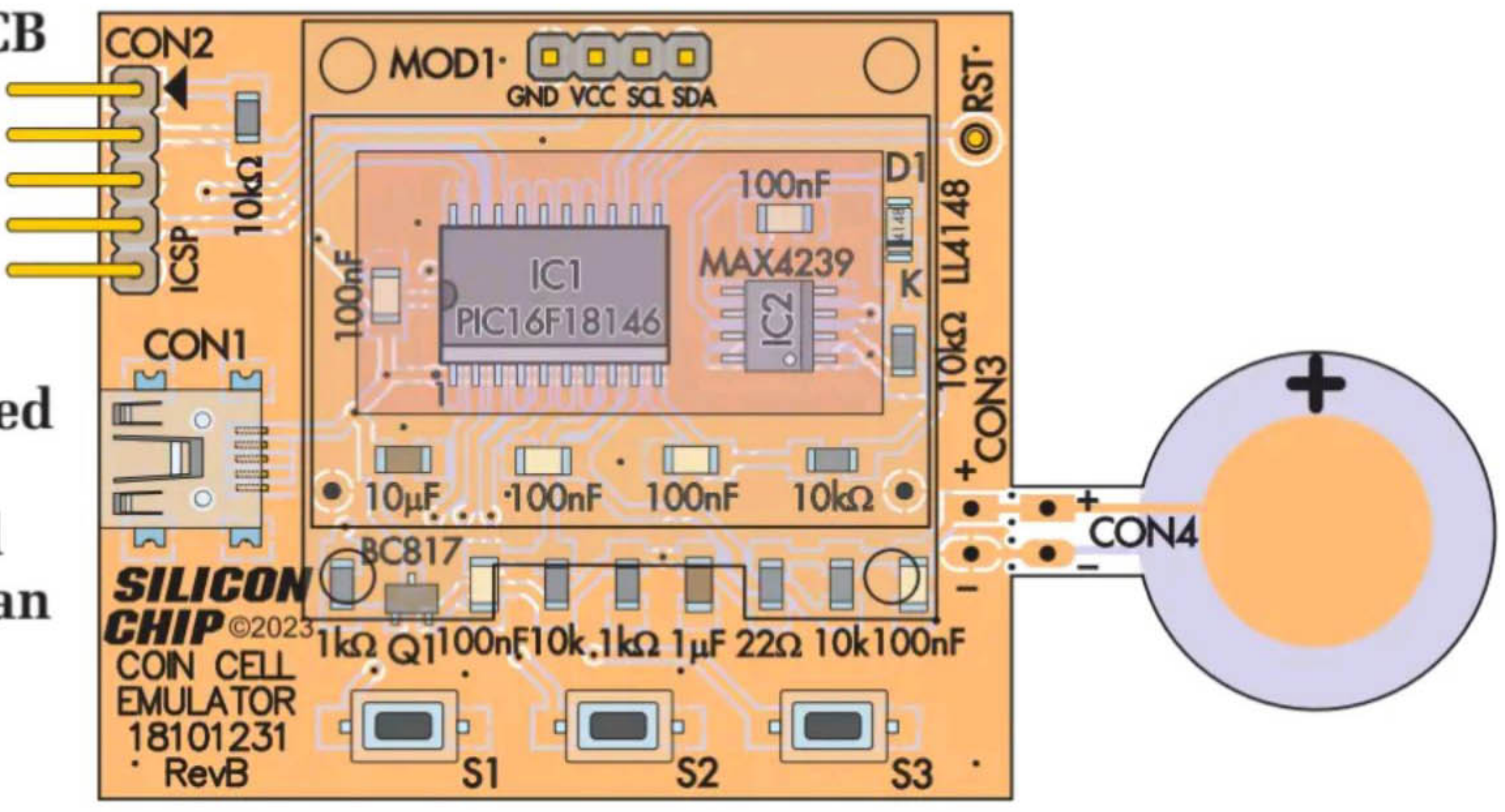
Assembly

The Emulator is built on a small PCB with surface-mounting components. They are the typical range of SOIC, SOT-23 and M3216/1206 parts that are fairly easy to solder. Fig.6 is the PCB overlay diagram; you can also refer to the photo of the PCB before the OLED module is attached.

We recommend using a fine-tipped soldering iron, solder flux paste, thin solder wire, tweezers, a magnifier and good lighting. Solder wicking braid is helpful for removing bridges and excess solder. Work outside if you don't have good ventilation or fume extraction.

Start with the mini-USB socket, CON1. Apply flux to all its pads and rest the part on top. Its locating pegs should lock into holes in the PCB, aligning it.

Fig.6: assembling the PCB mainly involves fitting SOIC and M3216/1206 SMD parts. Take care with the orientation of the two ICs and D1. 'Mousebites' are provided so you can separate the PCB between CON3 and CON4; the two halves can be rejoined with some light-duty figure-8 wire.



Clean your iron's tip and add a small amount of fresh solder, then touch it to where the pins meet the PCB pads. After that, apply a generous amount of solder to the four larger pads that affix the connector's shell.

If you have bridges between the pins, add some extra flux and press some fresh braid against the bridge with the iron.

When the braid has taken up solder, slowly draw both away together. If the part is flat against the PCB, surface tension should leave enough solder to form a solid joint.

Fit Q1 next by spreading flux on its PCB pads and resting it in place, being sure to align the body with the silkscreen printing. Tack one lead, ensure the part is flat and aligned within all pads, then solder the remaining leads.

Solder the two ICs next, using a similar process, starting with one lead to locate the part. Both ICs should have a small pin 1 divot in one corner, so align that with the PCB markings. For IC2, this might be a notch at the pin 1 end.

For diode D1, ensure its cathode stripe aligns with the 'K' marking on the PCB. After this, none of the components are polarised. The capacitors will not be marked, so be careful not to get them mixed up. The resistors will be marked with codes, as shown in the parts list.

The PCB will now need a thorough cleaning to remove flux residue. At the minuscule currents the Emulator measures, any contaminants can cause leakage and interfere with measurements.

Your flux might recommend a solvent, but we find that isopropyl alcohol works well (another great option is Chemtools Kleanium G2). Wipe away any excess solvent and allow the remainder to evaporate thoroughly.

Give the PCB a thorough check now that it has been cleaned, as any problems will be easier to spot and repair before the OLED is fitted, as it covers many of the components.

Now solder on the three tactile switches, being sure to align them within their silkscreen outlines and keep them flat against the PCB. If you need to program your microcontroller, add the CON2 ICSP header.

Next, solder the OLED module in place using its four-pin header, aligning the pin markings and spacing it above the other components on the PCB. When you are happy with its location, solder stiff wires to the lower corners of the OLED module and secure them to the through-hole pads in the PCB below.

Finally, attach the rubber feet to the underside of the PCB so it won't scratch your work surface.

Programming

PICs supplied in kits or purchased separately from the Silicon Chip Online

Parts List – Coin Cell Emulator

1 double-sided PCB coded 18101231, 78 × 44mm
1 Mini-USB SMD connector (CON1)
1 5-way right-angle male header, 2.54mm pitch (CON2; **optional**, for ICSP)
1 1.3in I²C blue OLED module (MOD1)
3 2-pin SMD tactile switches (S1-S3)
4 small self-adhesive rubber feet

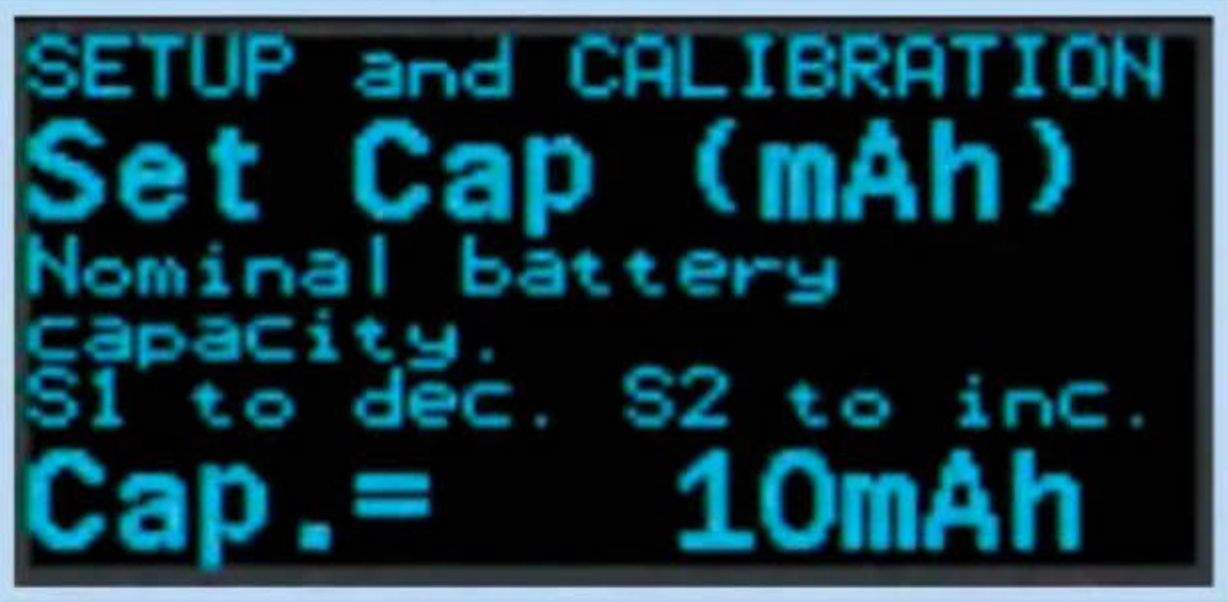
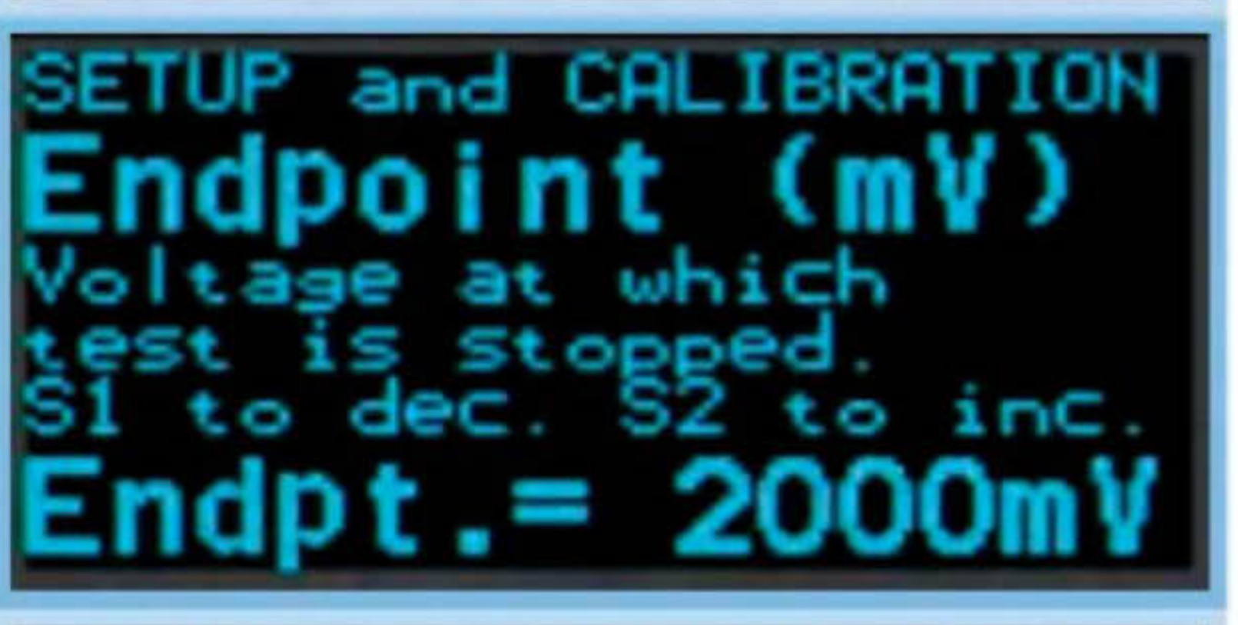
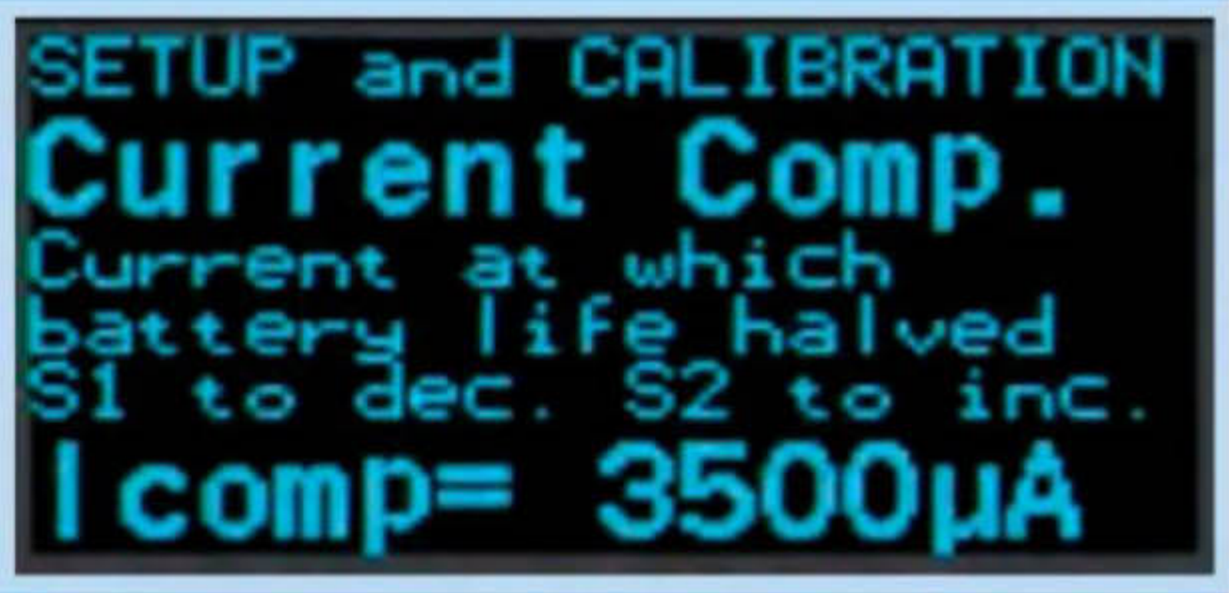
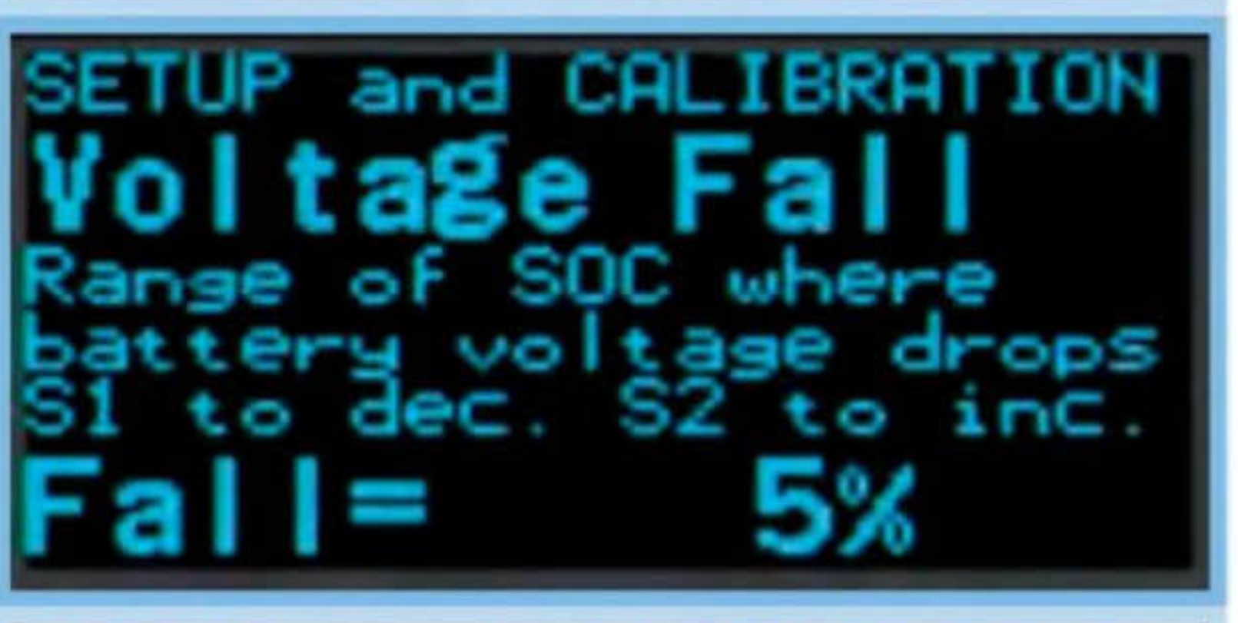
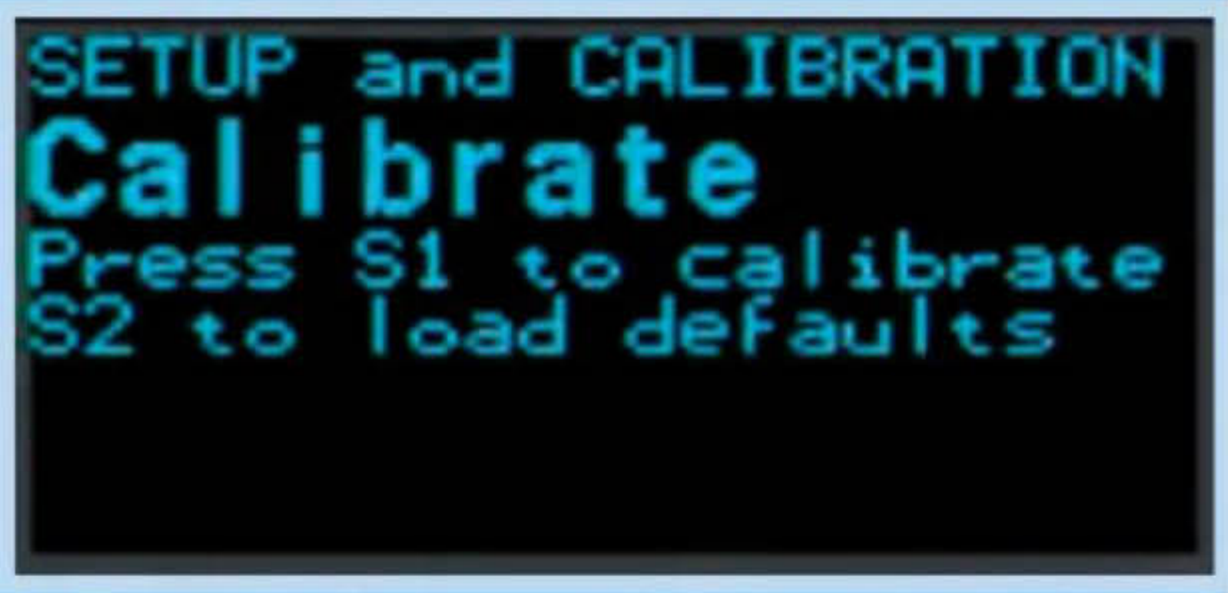
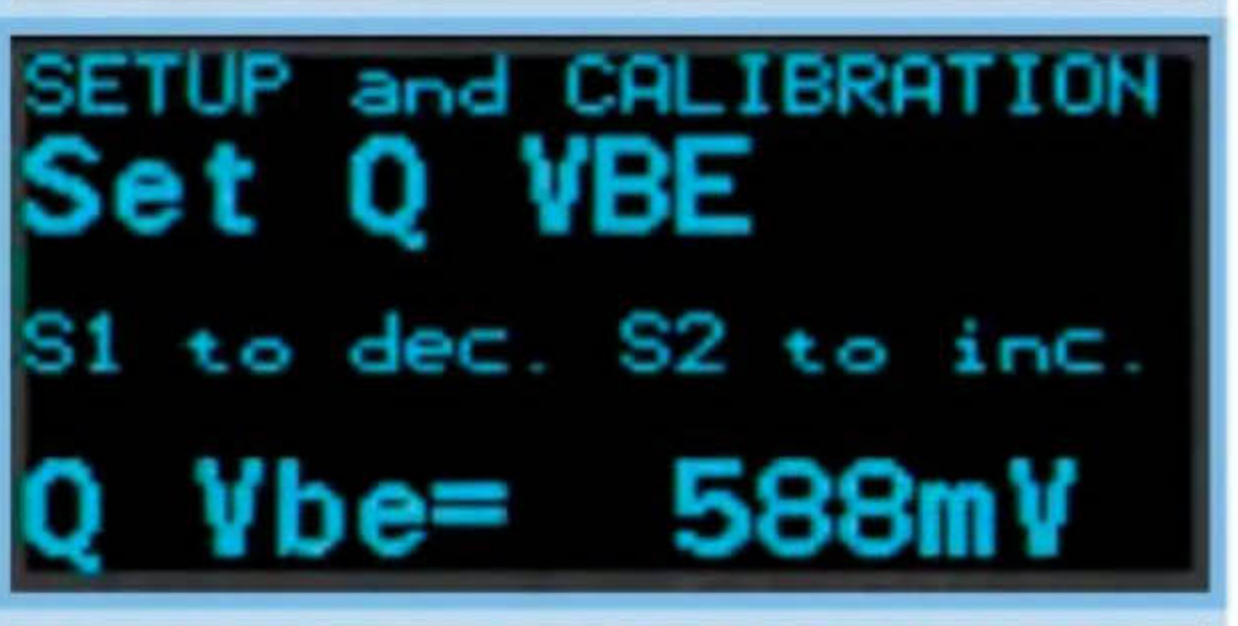
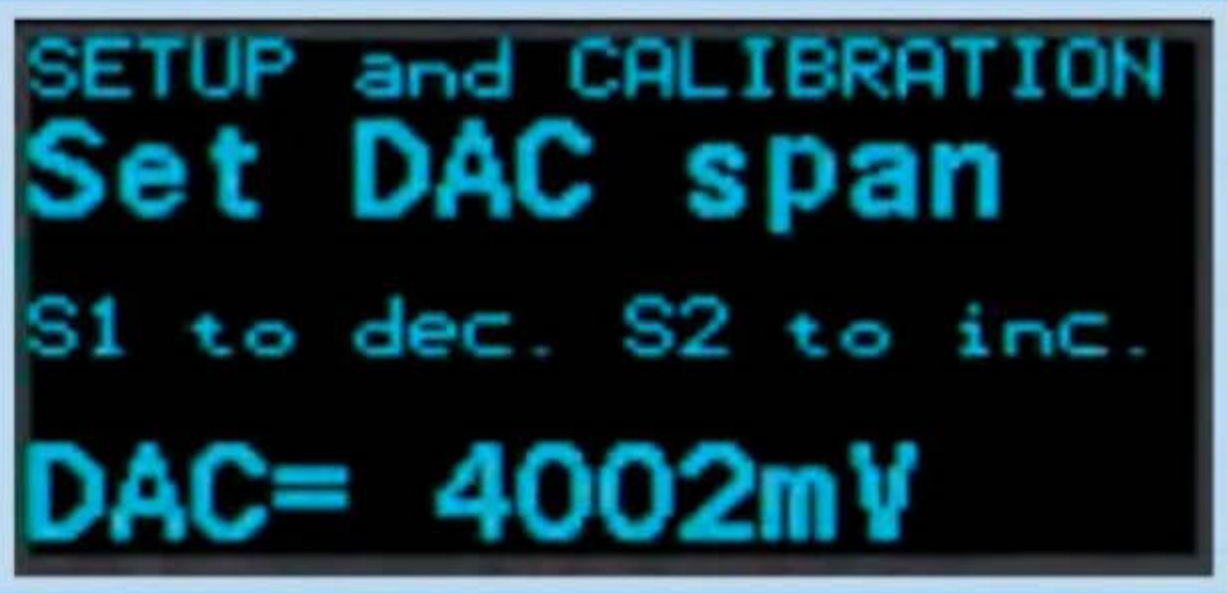
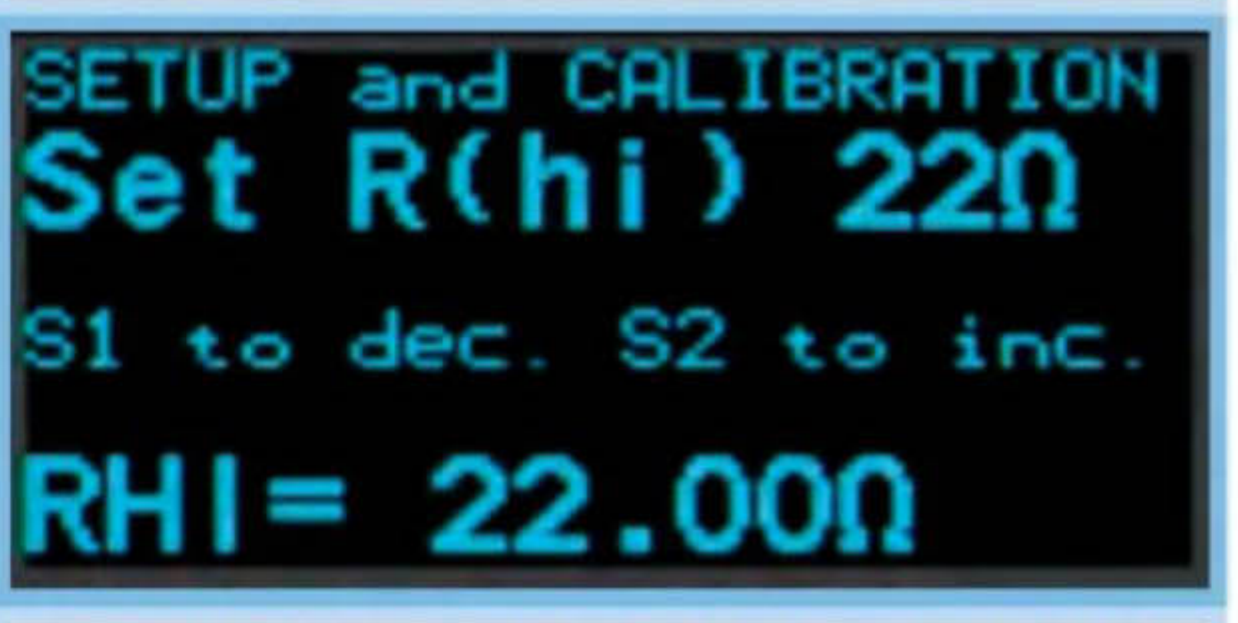
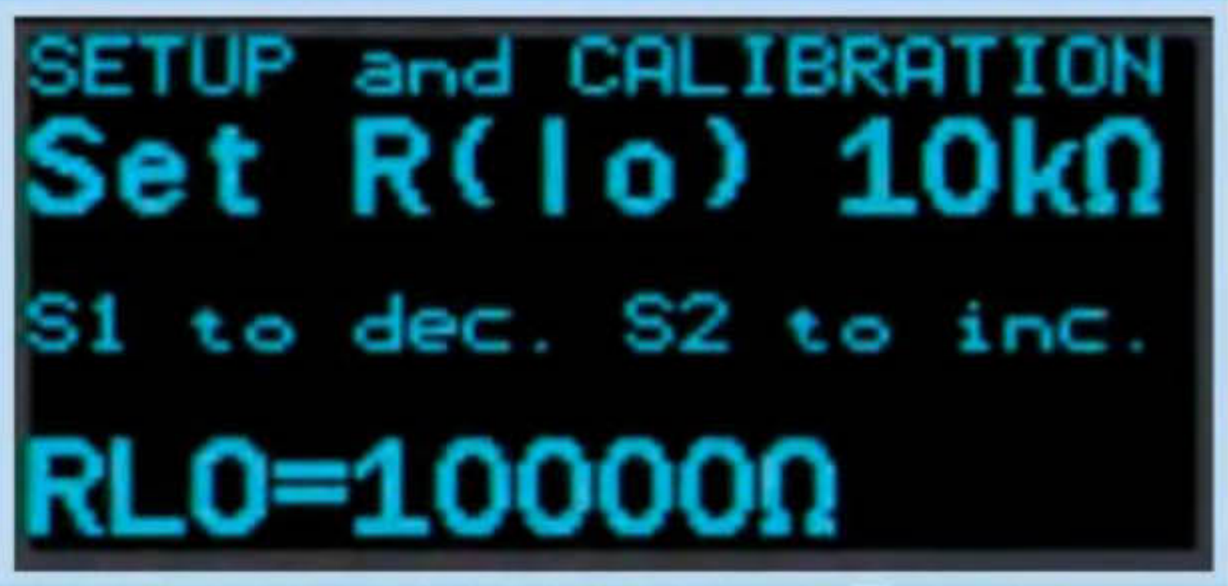
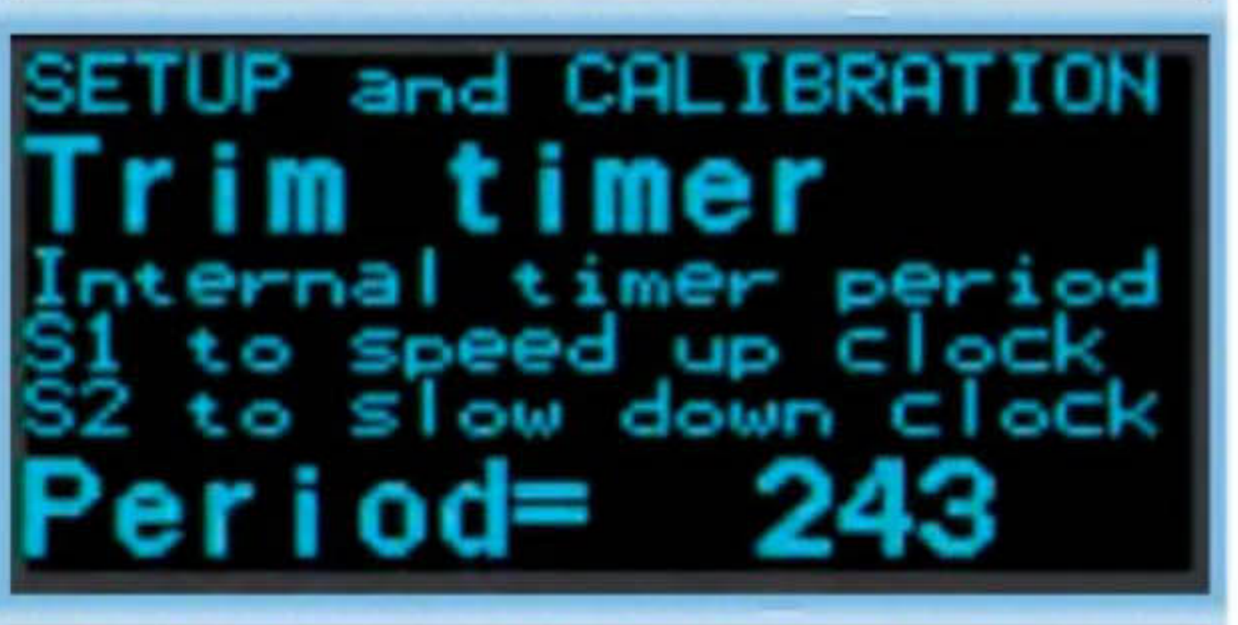
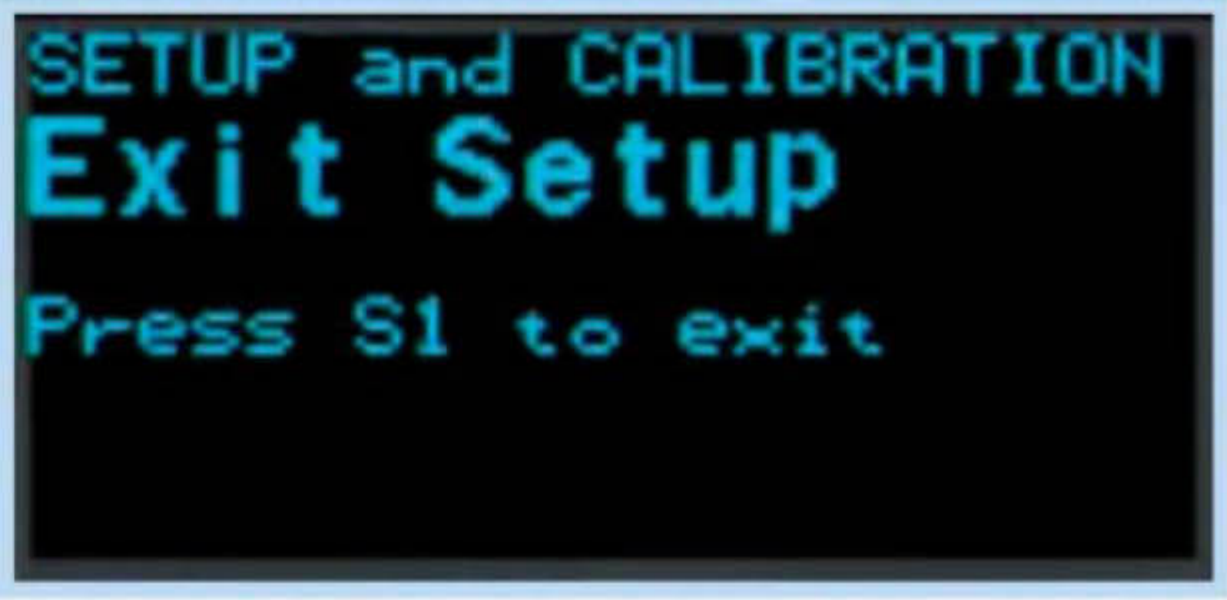
Semiconductors
1 PIC16F18146-I/SO microcontroller programmed with 1810123A.HEX, wide SOIC-20 (IC1)
1 MAX4238 or MAX4239 low-offset op amp, SOIC-8 (IC2)
1 BC817-40 NPN transistor, SOT-23 (Q1)
1 LL4148 SMD diode, SOD-80/MiniMELF (D1)

Capacitors *(all SMD M3216/1206 X7R)*
1 10µF 10V
1 1µF 16V
6 100nF 50V

Resistors *(all SMD M3216/1206 1% ¼W unless noted)*
5 10kΩ (code 1002 or 103)
2 1kΩ (code 1001 or 102)
1 22Ω ½W (code 22R0 or 22R)

Silicon Chip Coin Cell Emulator Kit
SC6823 (~£18 + P&P)
https://pemag.au/link/AC1J

Table 1 – Settings

Page – Parameter	**Notes**	**Screen**
Set Cap *Nominal emulated cell capacity* ***Default = 10mAh***	The default allows for brief tests. It can be set from 1-10mAh steps of 1mAh or up to 250mAh in steps of 5mAh.	 SETUP and CALIBRATION Set Cap (mAh) Nominal battery capacity. S1 to dec. S2 to inc. Cap.= 10mAh
Endpoint *The voltage at which tests stop* ***Default = 2V***	It can be set from 0 to 3.4V in 0.1V steps, the same as the output voltage.	 SETUP and CALIBRATION Endpoint (mV) Voltage at which test is stopped. S1 to dec. S2 to inc. Endpt.= 2000mV
Current Comp. *Determines how cell capacity is affected by high currents* ***Default = 3.5mAh***	The current at which the effective cell capacity is halved. It can be set in steps of 0.1mA; if set to 0mA, there is no compensation. 3.5mA is typical for CR2032 cells.	 SETUP and CALIBRATION Current Comp. Current at which battery life halved S1 to dec. S2 to inc. Icomp= 3500µA
Voltage Fall *SoC at which the cell voltage starts to decline* ***Default = 5%***	Below this level, the cell voltage setpoint is linearly decreased to reach 0V at 0% SoC. If set to 0%, then there is no decline in voltage.	 SETUP and CALIBRATION Voltage Fall Range of SOC where battery voltage drops S1 to dec. S2 to inc. Fall= 5%
Calibrate *Start automatic calibration voltage*	Ensure the output is not connected to any loads and press S1 to start. This sets the Q1 V_{BE} and DAC span automatically. Pressing S2 sets all parameters back to their defaults.	 SETUP and CALIBRATION Calibrate Press S1 to calibrate S2 to load defaults
Set Q1 V_{BE} *Transistor Q1 base-emitter junction voltage* ***Default = 588mV***	Set by the Calibrate step. If voltages across the range are still too high, increase this value. There is a slight offset below 0.3V output; voltages are not as accurate in that range.	 SETUP and CALIBRATION Set Q VBE S1 to dec. S2 to inc. Q Vbe= 588mV
Set DAC span *The nominal span of the DAC output* ***Default = 4002mV***	If the voltage offset increases across the range, decrease this; if it becomes lower, increase it.	 SETUP and CALIBRATION Set DAC span S1 to dec. S2 to inc. DAC= 4002mV
Set R(hi) (22Ω) *Actual value of 22Ω resistor* ***Default = 22.00Ω***	It can be set in steps of 0.01Ω within 10% of 22Ω. 1% parts should not need calibration.	 SETUP and CALIBRATION Set R(hi) 22Ω S1 to dec. S2 to inc. RHI= 22.00Ω
Set R(lo) (10kΩ) *Actual value of 10kΩ resistor* ***Default = 10000Ω***	It can be set in steps of 1Ω within 10% of 10kΩ. 1% parts should not need calibration.	 SETUP and CALIBRATION Set R(lo) 10kΩ S1 to dec. S2 to inc. RLO=10000Ω
Trim Timer *The displayed value is the period of the timer counter* ***Default = 243***	The Emulator's 1Hz clock is available at the RST pin (with respect to ground). This can be measured to help trim the timer. Each step will change the frequency by about 0.4%.	 SETUP and CALIBRATION Trim timer Internal timer period S1 to speed up clock S2 to slow down clock Period= 243
Exit Setup *Press S1 to return to normal operation*	All values are saved to EEPROM as soon as any changes are made and new settings are used immediately.	 SETUP and CALIBRATION Exit Setup Press S1 to exit

Shop come programmed, so skip this section if you have one of those.

The PIC16F18146 requires a PICkit 4, PICkit 5 or Snap programmer. If you are using a Snap (which does not provide power), you can supply power using a USB cable connected to CON1. You might need to use some short extension wires to prevent the Snap from fouling the USB cable.

You can use the Microchip IPE to program the 1810123A.HEX file. If you don't have the IPE installed, it can be downloaded and installed for free as part of the most recent MPLAB X IDE.

Once programmed, the startup OLED screen should look like Screen 1.

Setup

The Coin Cell Emulator is usable without calibration, but we recommend doing it since it is easy and only needs to be done once. Hold in S3 until the screen goes blank, then release it to enter SETUP mode.

Table 1 summarises the individual setup pages you can cycle through by pressing S3. In general, S1 decreases a parameter while S2 increases it. On some pages, they trigger specific actions, such as starting the automatic calibration process or returning to normal operation from SETUP.

The first four SETUP screens relate to the emulation settings and can be skipped to reach the calibration settings.

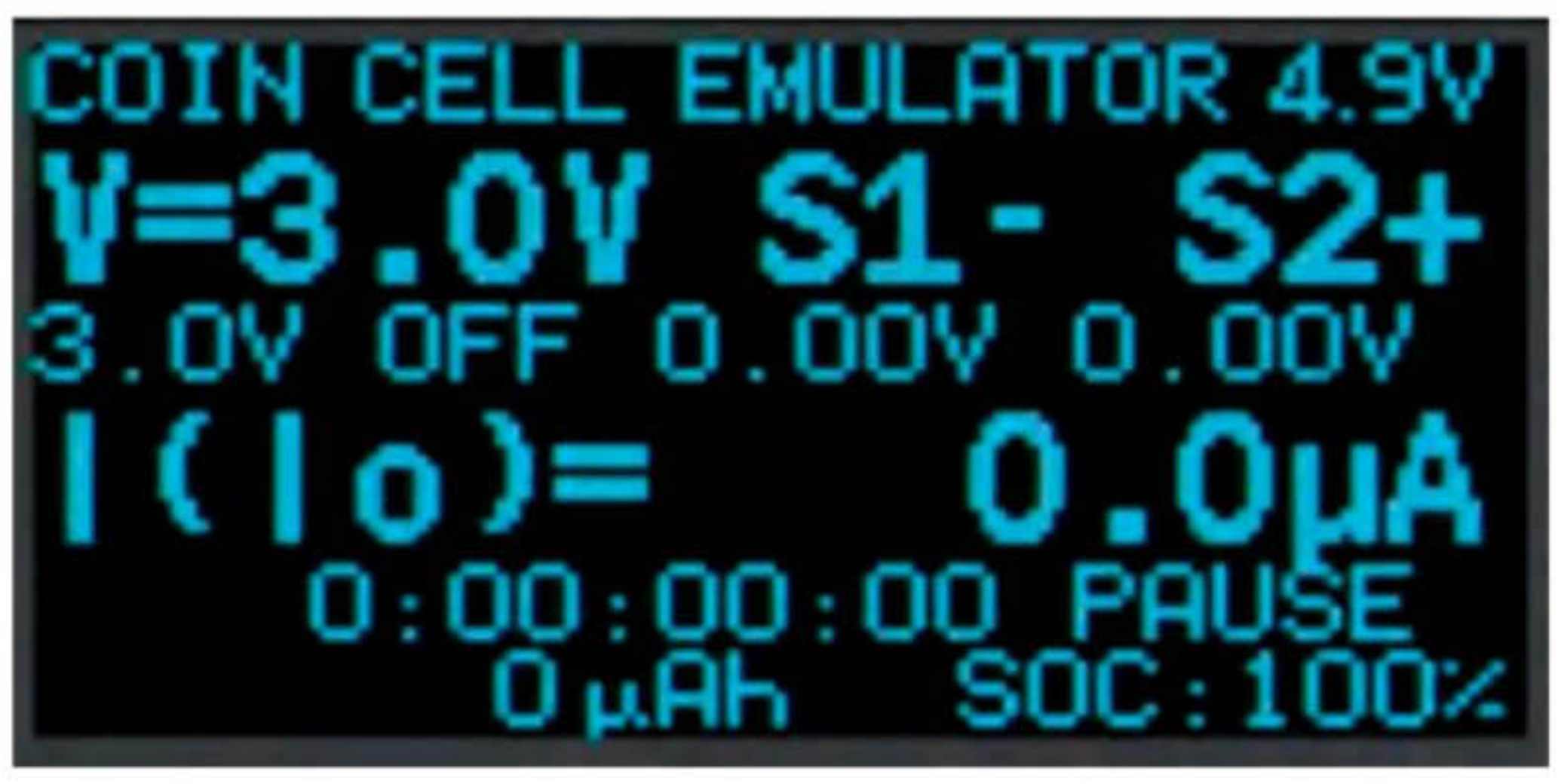
We recommend just running the automatic "Calibrate" step. If the Emulator's other measurements are off, you could consider changing other values, such as the resistances or timer trim. Cycle to the Exit Setup page and press S1 to return to regular operation.

Connections

CON3, CON4 and the circular pads can all be used to connect to a circuit under test. For most of our prototyping, we simply used a header socket for CON3 and ran jumper wires to our circuit.

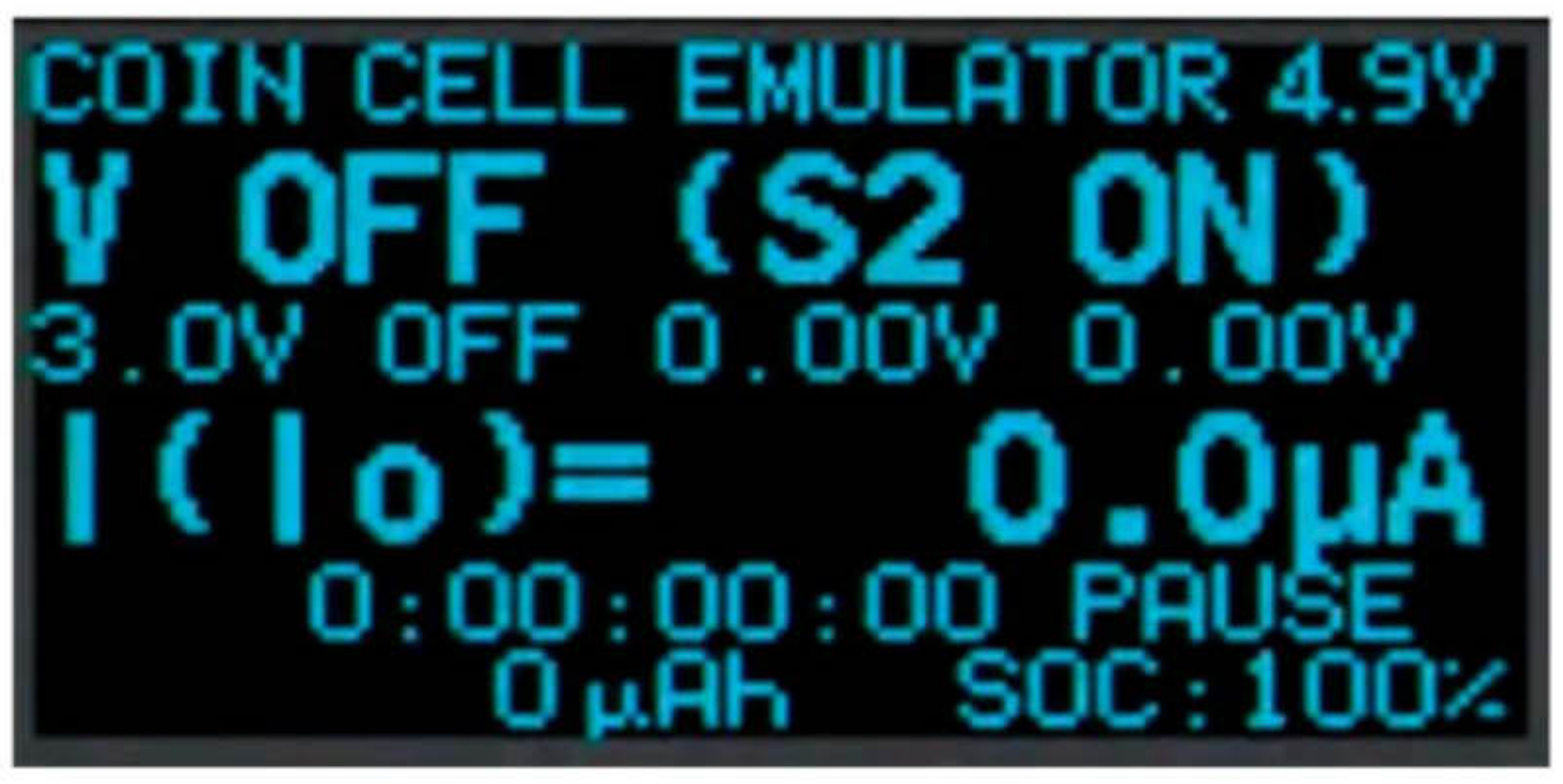
The circular section of the PCB is designed to be slotted into the side of a cell holder. The photo opposite shows the Emulator connected to our Advanced Test Tweezers. It probably won't work with other cell holder types where the cell is inserted from above.

Since the PCB is only 1.6mm thick, it will not be a tight fit for holders that expect a 3.2mm-high CR2032



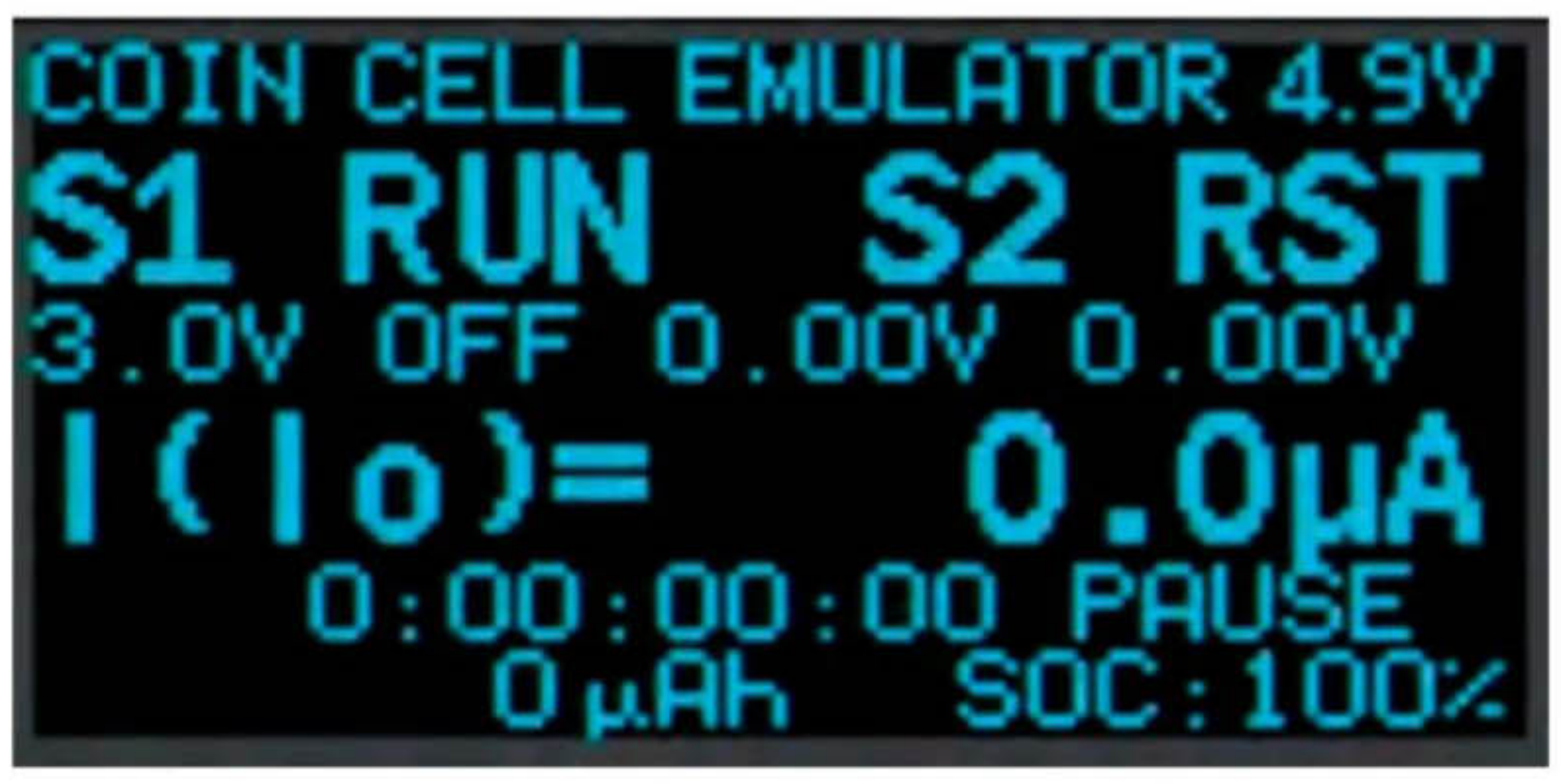
```
COIN CELL EMULATOR 4.9V
V=3.0V S1- S2+
3.0V OFF 0.00V 0.00V
I(lo)=    0.0µA
   0:00:00:00 PAUSE
     0µAh   SOC:100%
```

Screen 1: the initial screen seen when the Emulator powers on allows the output voltage setpoint to be changed with pushbuttons S1 and S2. S3 switches to the other screens. Holding S3 for three seconds enters the Setup mode, shown in Table 1.



```
COIN CELL EMULATOR 4.9V
V OFF (S2 ON)
3.0V OFF 0.00V 0.00V
I(lo)=    0.0µA
   0:00:00:00 PAUSE
     0µAh   SOC:100%
```

Screen 2: the output can be toggled on and off when this screen is shown. Note also the supply voltage display at upper right. If this is flashing, the supply is lower than 4.5V or higher than 5.5V, and the Emulator may not function correctly.



```
COIN CELL EMULATOR 4.9V
S1 RUN  S2 RST
3.0V OFF 0.00V 0.00V
I(lo)=    0.0µA
   0:00:00:00 PAUSE
     0µAh   SOC:100%
```

Screen 3: S1 and S2 start and reset the stopwatch timer and charge accumulator measurement, respectively. If the timer is running, this screen will show PAUSE instead, with S1 pausing the timer if pressed.

cell, although many holders are designed to accept 1.6mm thick CR2016 cells. You could carefully bend the cell holder's tabs to add more tension. We also added some thickness to the Emulator by soldering on some pieces of wire, as shown on page 56.

Another option is to carefully break the PCB between CON3 and CON4 (there are 'mouse bites' in the PCB to facilitate this). You could then run a pair of wires between CON3 and CON4 to join them.

Operation

Screen 1 shows the default Emulator cell voltage of 3V, which can be changed on that page. Other features on Screen 1 are common to the operating Screens. The third line of text shows the status of the output voltage; the first figure is the setpoint (target) output voltage and whether it is on or off.

The other voltages are the values upstream and downstream of the shunt, respectively. They can be considered the internal cell voltage and external 'terminal' voltage, respectively. The first should be very close to the set voltage (when on), except if the emulated cell is nearly flat.

The fourth line (in larger text) shows the measured current. It is in a larger font as it is the most important parameter to observe. If "I(lo)" is shown, the reading is expected to be accurate to 0.1µA as only the 10kΩ resistor is being used as a shunt.

When "I(hi)" is shown, the Emulator has switched to the higher range and the 22Ω resistor comes into play. When this happens depends on the output voltage and supply voltage (which relates to IC2's headroom). At 3V output, it will occur at around 130µA.

The second-last line shows the stopwatch timer, which measures up to 999 days, or almost three years. The text on this line indicates if the timer is running and, if so, the charge measurements on the next line are also accumulating.

The µAh reading on the last line measures actual charge consumption (not adjusted). It can be used to validate the total current consumption and estimate potential capacity losses due to high current usage. The SoC figure does take into account the adjusted current.

Pressing S3 shows Screen 2, which allows the output voltage to be switched on and off; S1 switches it off, while S2 switches it on.

Screen 3 is reached by pressing S3 again; it allows the timer and charge accumulator to be paused, started and reset. S1 will start and pause the timer, while pressing S2 resets the timer and accumulator when the timer is paused.

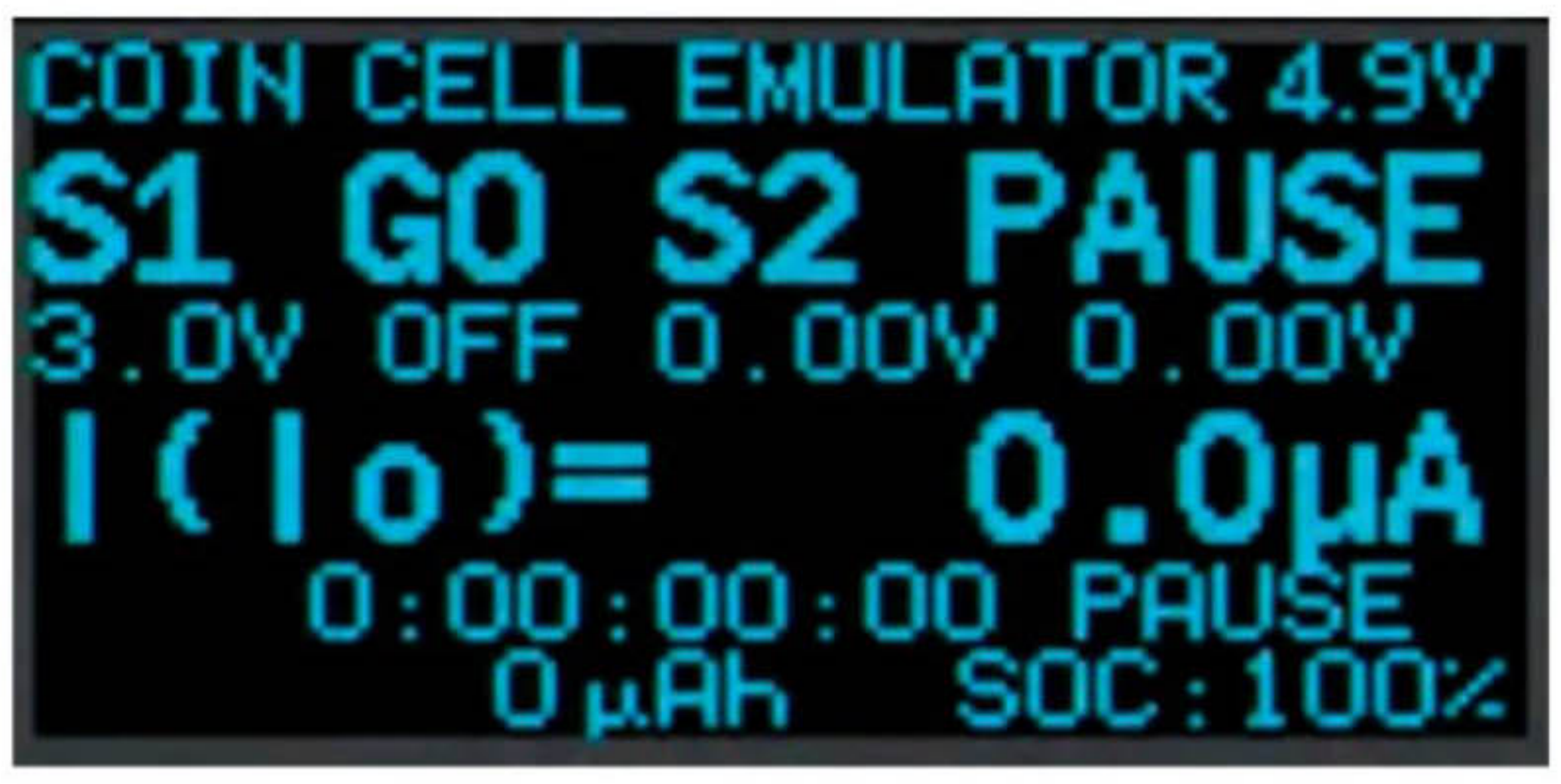
Press S3 again to reach Screen 4. Pressing S1 ("GO") on this screen will switch on the output voltage and start the timer and accumulator; S2 ("PAUSE") will pause the timer and switch the output off. Thus it can be used to start and stop testing cycles.

Once you've started a test, the current draw will be shown, and the timer and accumulator will go up while the SoC goes down.

As the SoC passes 5%, the output voltage will drop to simulate the cell running flat. When the output voltage reaches the endpoint, the test will pause, as if S2 were pressed on this screen, allowing the statistics to be recorded.

Conclusion

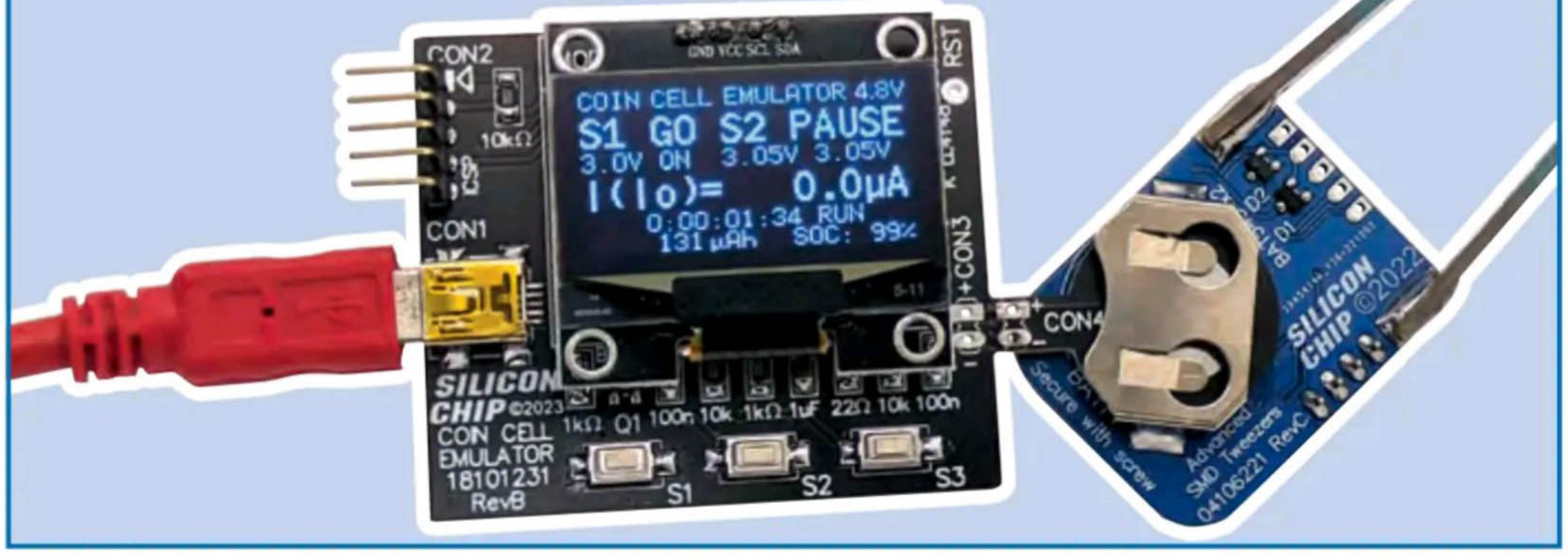
We're already making good use of the Coin Cell Emulator in designing an upcoming project. It's also coming in handy as a general power supply for low voltages and currents. PE



```
COIN CELL EMULATOR 4.9V
S1 GO S2 PAUSE
3.0V OFF 0.00V 0.00V
I(lo)=    0.0µA
   0:00:00:00 PAUSE
     0µAh   SOC:100%
```

Screen 4: pressing S1 here starts the timer and charge accumulator and switches on the output voltage. S2 pauses the test, allowing the results to be recorded. The test will be automatically paused if the Emulator reaches its endpoint voltage.

The circular section of the PCB is designed to slot straight into the cell holder we've used for various projects, including the Advanced Test Tweezers shown here. In this case, testing would be easier if we separated the PCB between CON3 and CON4 for a more flexible connection.



WIRELESS FOR THE WARRIOR

by **LOUIS MEULSTEE**

THE DEFINITIVE TECHNICAL HISTORY OF RADIO COMMUNICATION EQUIPMENT IN THE BRITISH ARMY

The *Wireless for the Warrior* books are a source of reference for the history and development of radio communication equipment used by the British Army from the very early days of wireless up to the 1960s.

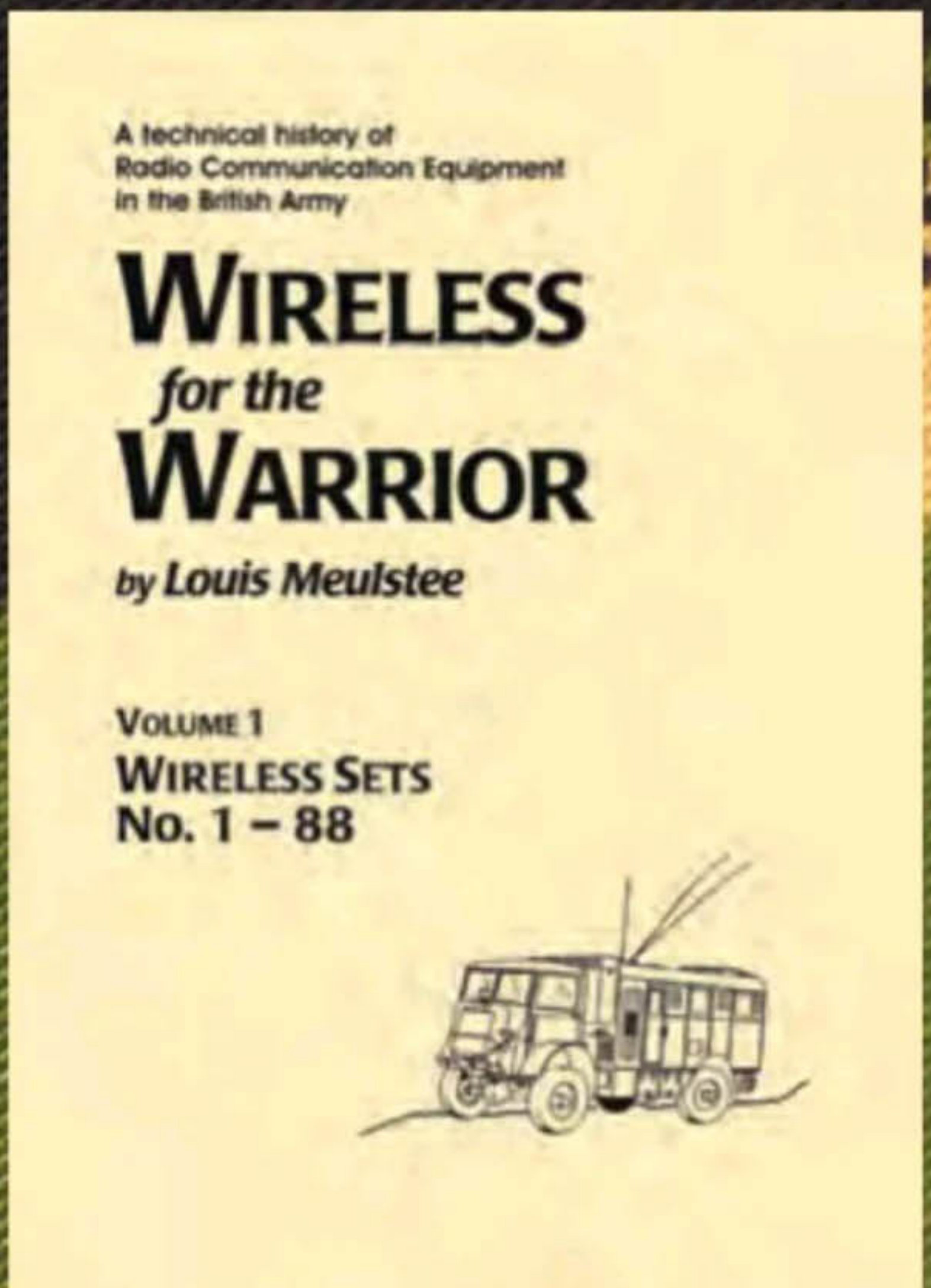
The books are very detailed and include circuit diagrams, technical specifications and alignment data, technical development history, complete station lists and vehicle fitting instructions.

***Volume 1* covers transmitters and transceivers used between 1932-1948. This era starts with positive steps to develop a new series of wireless sets that offered great improvements over World War I equipment. The other end of this timeframe saw the introduction of VHF FM and hermetically sealed equipment.**

***Volume 2* was a continuation of Volume 1 but is unfortunately now out of print.**

***Volume 3* covers army receivers from 1932 to the late 1960s. The book not only describes receivers specifically designed for the British Army, but also the Royal Navy and RAF. Also covered: special receivers, direction finding receivers, Canadian and Australian Army receivers, commercial receivers adopted by the Army, and Army Welfare broadcast receivers.**

***Volume 4* covers clandestine, agent or 'spy' radio equipment, sets which were used by special forces, partisans, resistance, 'stay behind' organisations, Australian Coast Watchers and the diplomatic service. Plus, selected associated power sources, RDF and intercept receivers, bugs and radar beacons.**

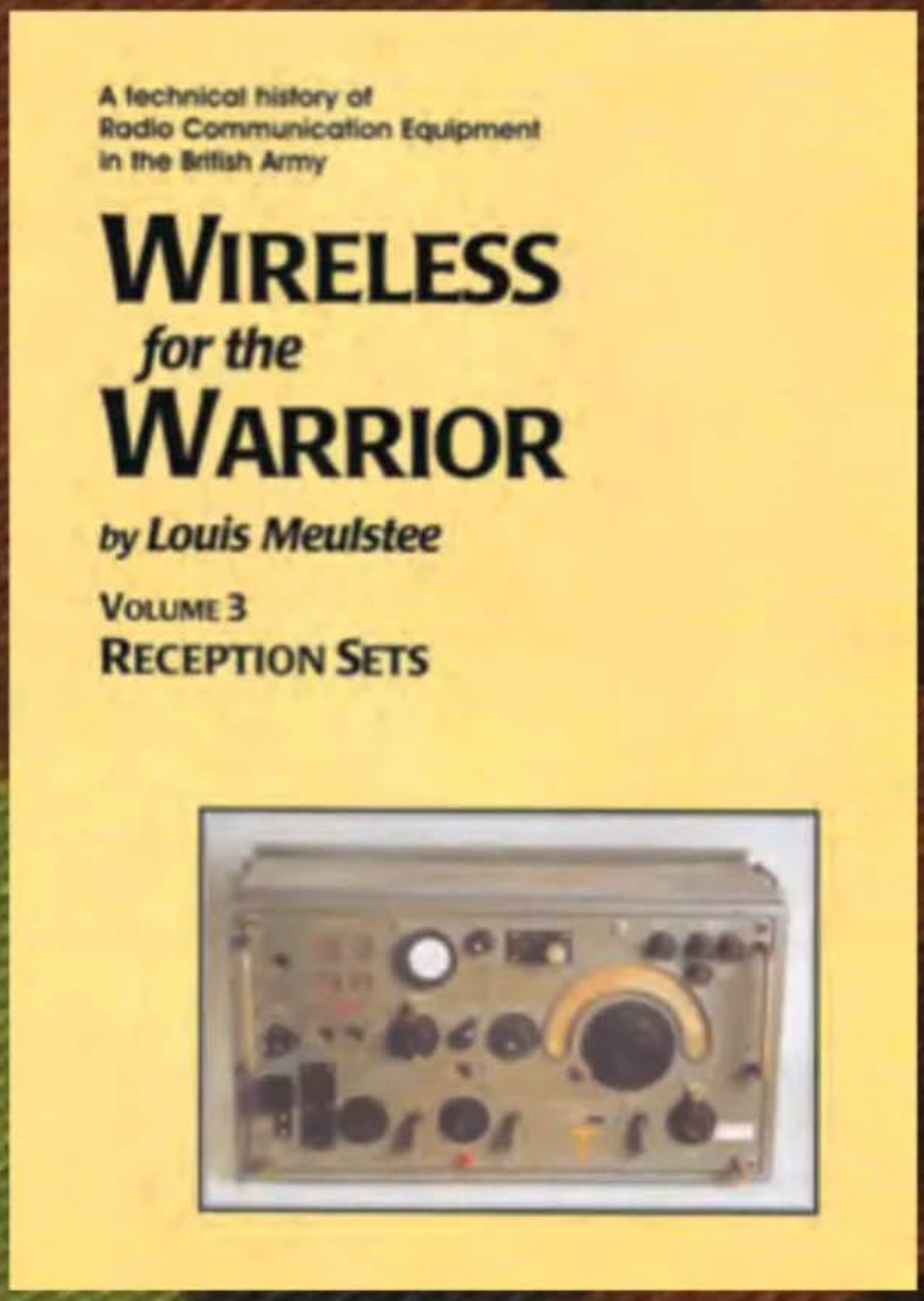


A technical history of
Radio Communication Equipment
in the British Army

WIRELESS
for the
WARRIOR

by Louis Meulstee

VOLUME 1
WIRELESS SETS
No. 1 – 88

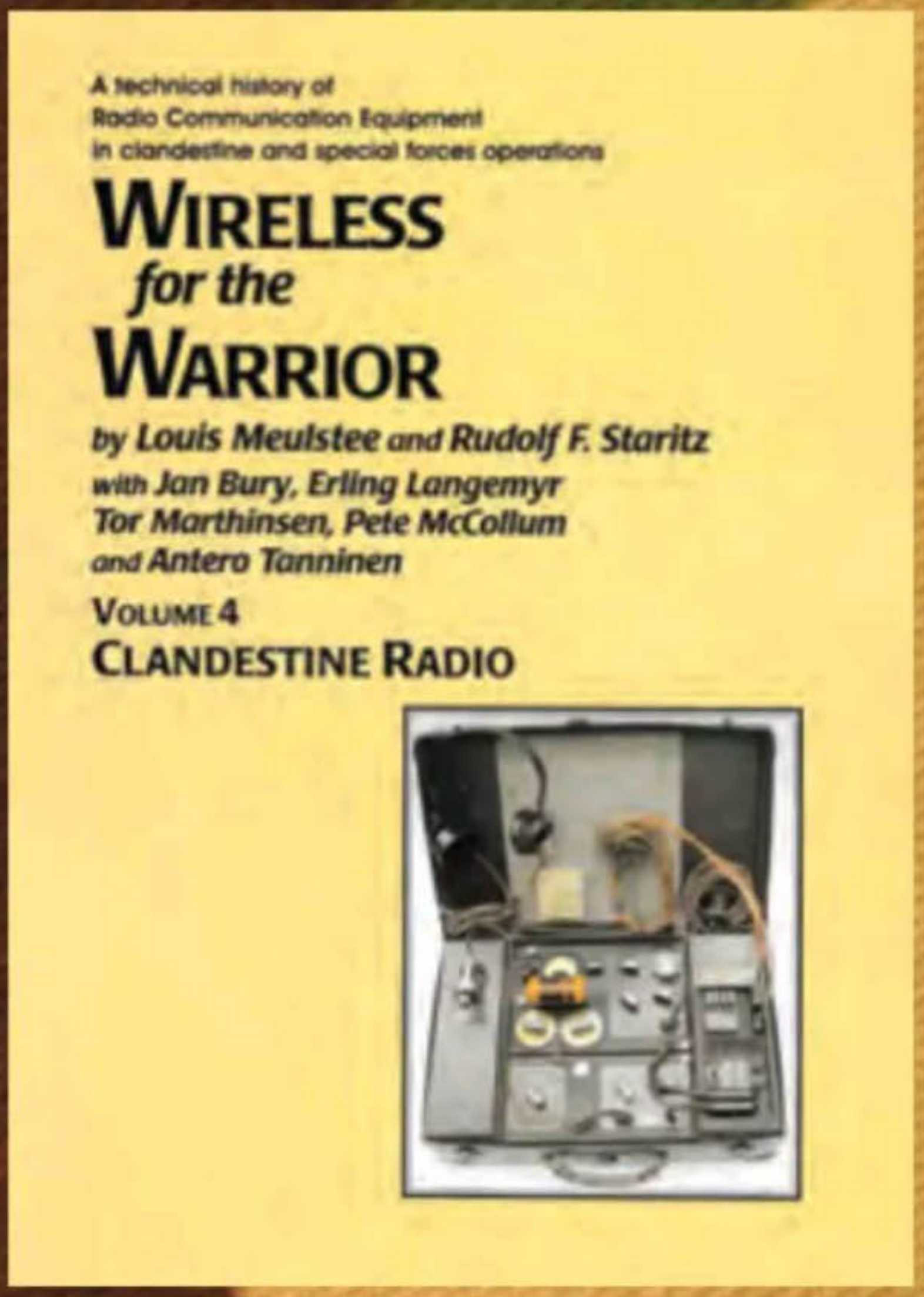


A technical history of
Radio Communication Equipment
in the British Army

WIRELESS
for the
WARRIOR

by Louis Meulstee

VOLUME 3
RECEPTION SETS



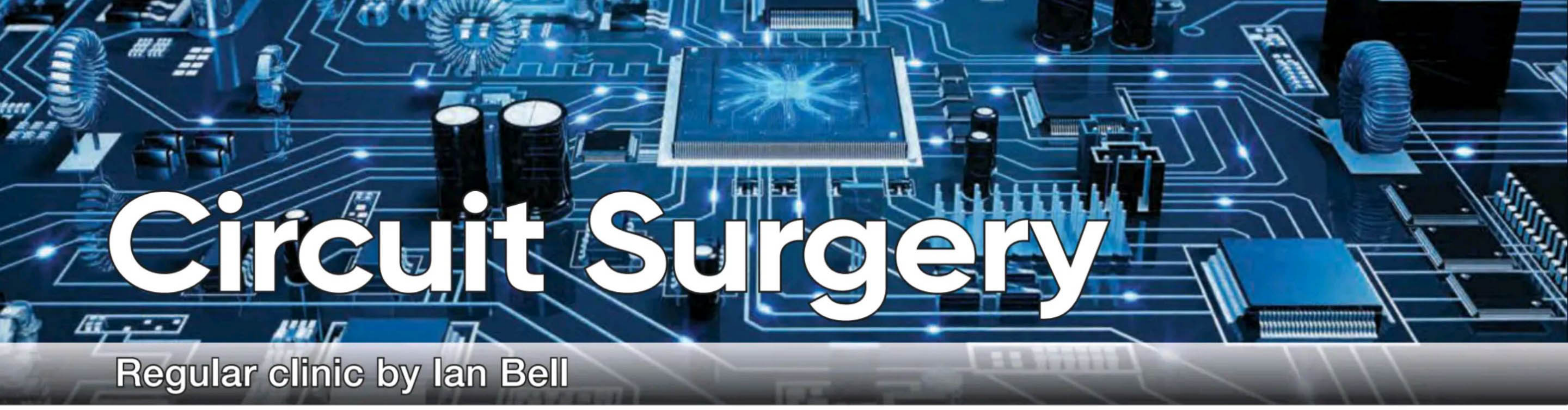
A technical history of
Radio Communication Equipment
in clandestine and special forces operations

WIRELESS
for the
WARRIOR

by Louis Meulstee and Rudolf F. Staritz
with Jan Bury, Erling Langemyr
Tor Marthinsen, Pete McCollum
and Antero Tanninen

VOLUME 4
CLANDESTINE RADIO

ORDER YOURS TODAY!
JUST CALL 01202 880299 OR VISIT www.electronpublishing.com



Circuit Surgery

Regular clinic by Ian Bell

Topics in digital signal processing – digital-to-analog converter (DAC) frequency response

We are looking at various topics related to digital signal processing (DSP).

DSP covers a wide range of electronics applications where signals are manipulated, analysed, generated, stored or displayed as digital data but originate from and/or are converted to real-world signals for interaction with humans or other parts of the physical world.

Fig.1 shows the key elements of a generic DSP system with a signal path from an analog input via digital processing to an analog output. This does not necessarily represent every DSP system (not all have all the parts shown), but it serves are as reference for the various subsystems we will consider.

Last month, we discussed the properties of the DAC output signal and the reconstruction filters required on DAC outputs.

We described the LTspice behavioural sample-and-hold block that can be used to generate waveforms like those produced by DACs, for example, to use as input for simulation of reconstruction filters.

We also discussed the fact that a DAC does not have a flat frequency response, which may need to be addressed in some DSP systems. We will continue with that theme this month.

DAC frequency response recap

As discussed last month, at first glance, it may seem that if we have a very good DAC operating at the original sampling frequency, a sufficient sampling rate (for the signal frequencies used) and a reconstruction filter with a flat passband and a sufficiently sharp cutoff, we can more-or-less perfectly reconstruct a sampled signal.

However, this overlooks the fact that the shapes of an idealised sample waveform (a train of impulses) and the output from the DAC (stepped waveform) are not the same. That implies the DAC does not have a flat frequency response, which in turn means that even an ideal DAC and flat-passband brickwall lowpass reconstruction filter will not perfectly reproduce the sampled waveform.

The shape of a DAC's frequency response relates to the sinc mathematical function, which is defined as:

$$\mathrm{sinc}(x)=\frac{\sin(x)}{x}$$

sinc(0) is defined to be 1. The sinc function occurs frequently in DSP and telecommunications, where it is commonly written terms of πx rather than x. The frequency response (variation of gain with frequency, f) of a DAC is given by:

$$\mathrm{abs}\left(\frac{\sin(\pi f/f_s)}{\pi f/f_s}\right)$$

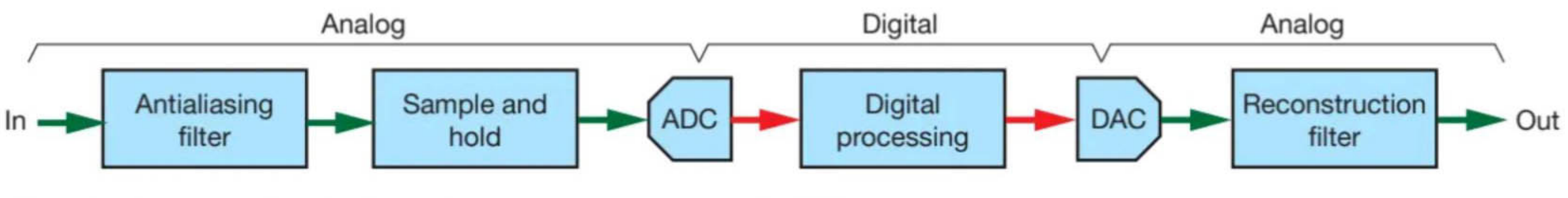


Fig.1: A generic digital signal processing (DSP) system structure.

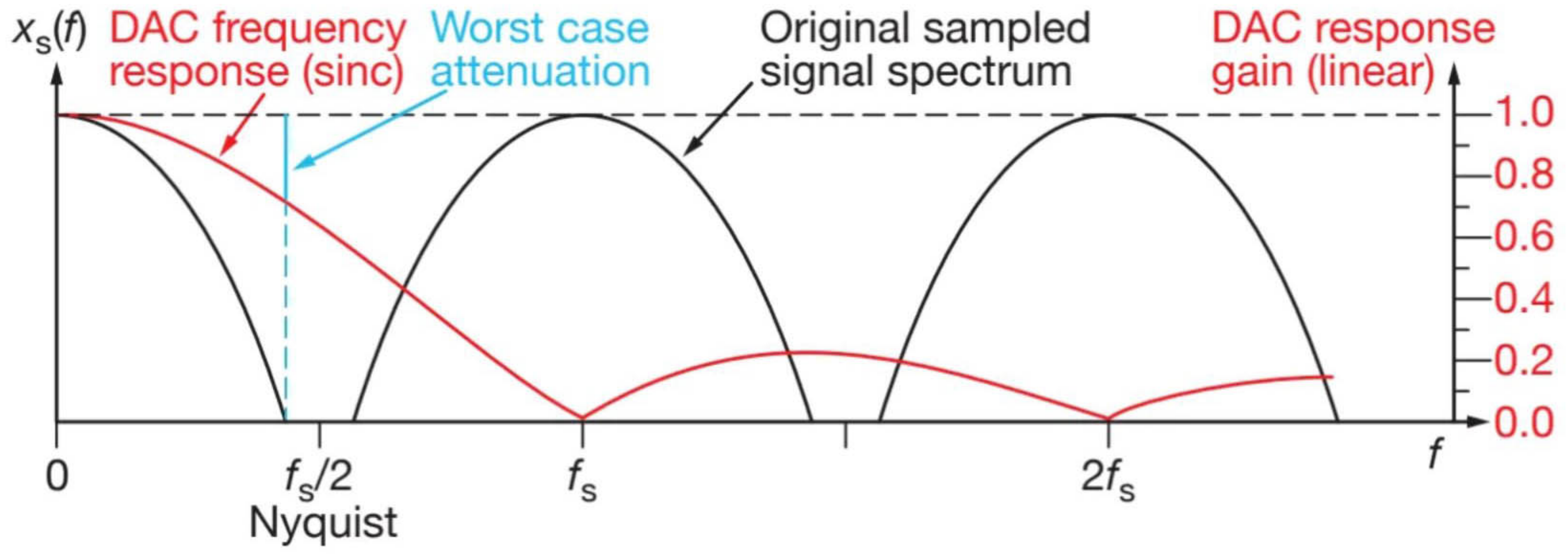


Fig.2: The DAC frequency response (sinc function) with a generic sampled signal spectrum, using a linear scale for the DAC response.

f_s is the sampling frequency and abs(x) is the absolute value of x (ie, $-x$ [making a negative x positive] if $x < 0$, otherwise x; also written $|x|$).

Fig.2 shows the DAC frequency response superimposed on a generic (and arbitrarily shaped) sampled signal spectrum. The plot shows how the DAC response aligns with the signal spectrum.

Over the wanted range (up to $f_s/2$), the effect of the DAC's frequency response is to attenuate the higher signal frequencies. To have no effect, it would have to be flat up to $f_s/2$ which is the maximum frequency of the wanted (originally sampled) signal. Note that the DAC's gain is zero at integer multiples of the sampling frequency.

Does it matter; what can we do?

There are several ways of dealing with a DAC's sinc-shaped frequency response. The simplest is to ignore the problem. Although it is not very obvious in Fig.2, the impact of the DAC's sinc-shaped frequency response is relatively small at lower frequencies. sinc(x) is close to 1 for small values of x, which corresponds frequencies well below the sampling frequency (where the term f/f_s is small).

In some systems, there is relatively little content in frequencies approaching $f_s/2$, either because it is not present to a significant extent in the input signal, or because it has been removed by input filtering. In some cases, the acceptable level of signal fidelity does not merit the extra effort of dealing with the sinc response.

In systems where the sinc shape of the DAC frequency response is a problem, a few approaches can be used to mitigate it.

It is possible to correct for the DAC response using a filter with an equal and opposite frequency response to the DAC.

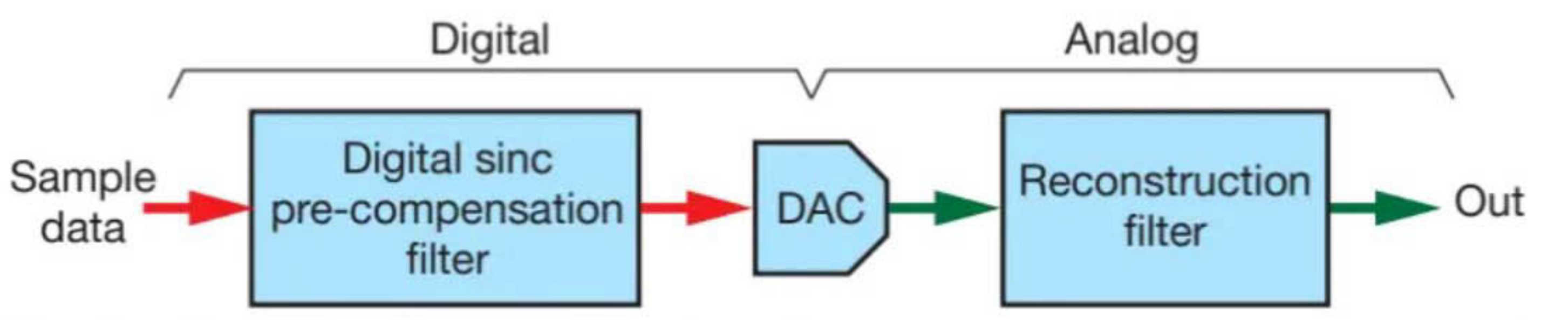


Fig.3: digital pre-equalisation frequency response correction.

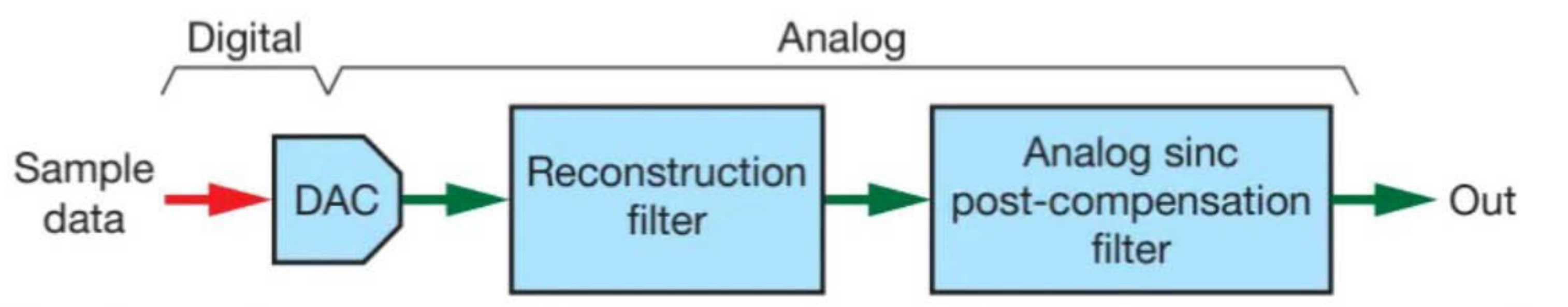


Fig.4: analog post-equalisation frequency response correction.

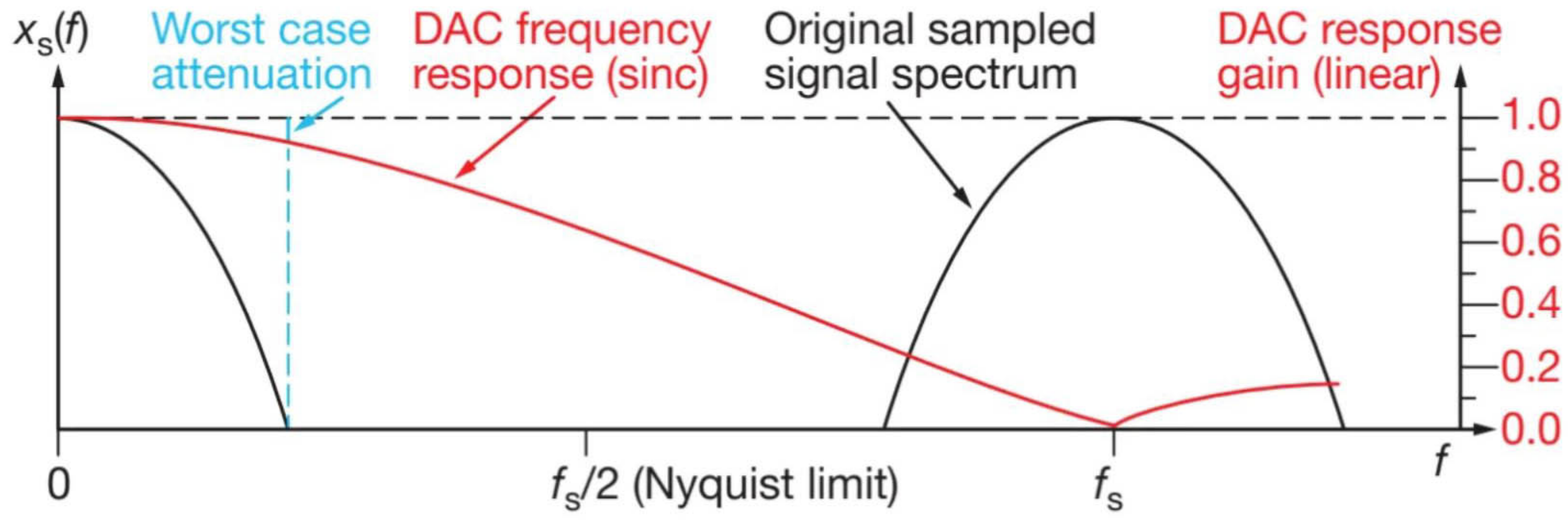


Fig.5: the same signal as in Fig.2 but sampled at a higher frequency, with a consequent reduction in DAC attenuation.

This can be achieved using a digital filter before the DAC (referred to as a sinc pre-compensation or pre-equalisation filter), as shown in Fig.3.

Alternatively, it can be achieved using an analog filter after the DAC and reconstruction filter (post-compensation or post-equalisation), as shown in Fig.4. These filters can flatten the frequency response but also worsen the signal-to-noise ratio at low frequencies.

Sampling rate and interpolation

As noted previously, the effect of the DAC's frequency response is worse at lower sampling rates relative to the original signal, so increasing the sampling rate will mitigate the effects of the DAC's frequency response. The cyan line in Fig.2 highlights the worst attenuation due to the DAC frequency response, which occurs at the highest signal frequency.

The original signal depicted in Fig.2 has a low amplitude at this frequency, but this is just a representative shape – the signal amplitude could easily be very significant at this frequency, implying a potentially large impact from the sinc attenuation.

Fig.5 shows the same original signal sampled at a higher frequency. The relative shapes of the signal spectra and DAC response in this example result in much lower worse-case attenuation than for the situation in Fig.2.

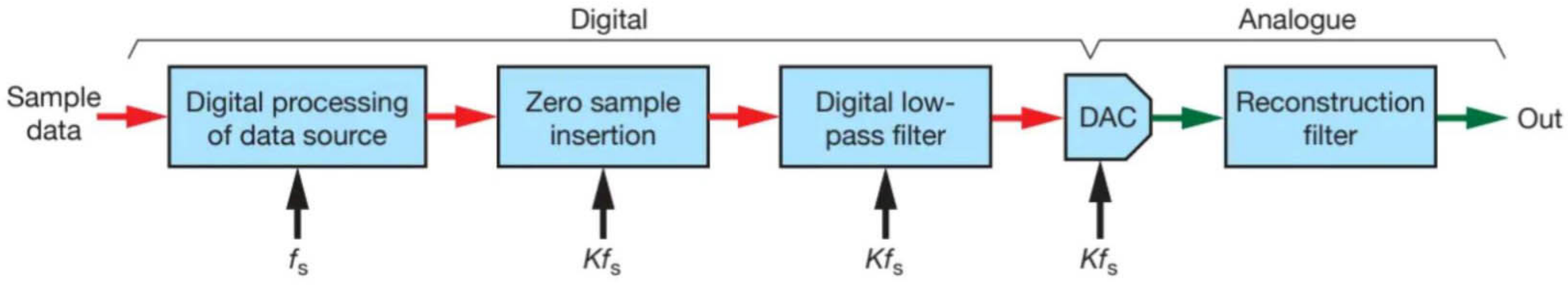


Fig.6: a block diagram of an interpolating (or 'oversampling') DAC.

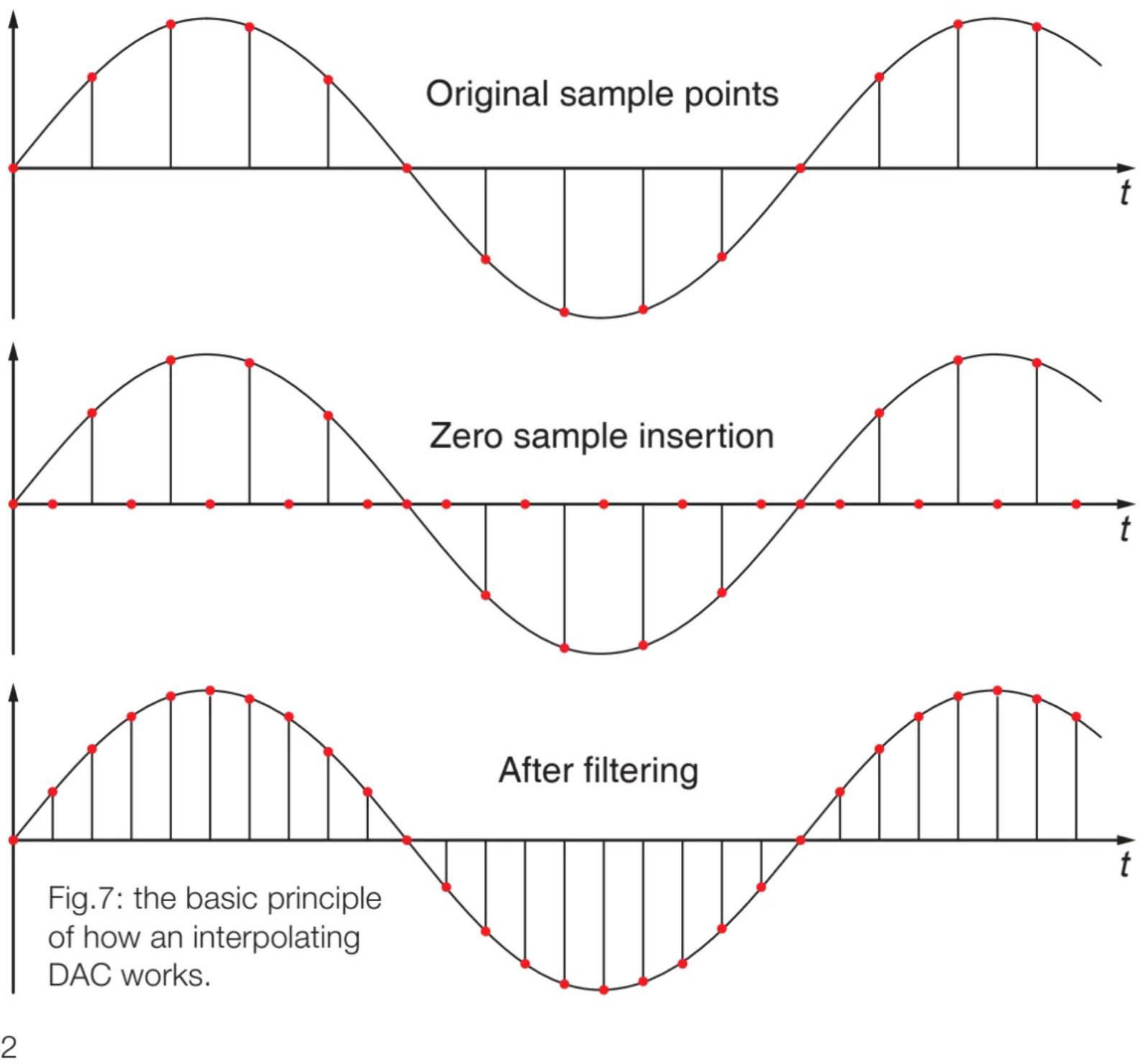


Fig.7: the basic principle of how an interpolating DAC works.

This scenario also puts less demand on the reconstruction filter. There is large gap between the wanted signal and the first spectral image of the sampled signal, so the reconstruction filter can roll off more slowly and achieve the same signal quality as a faster cutoff filter applied in the situation shown in Fig.2.

The obvious approach to increasing the sampling frequency of the DAC may be to run the whole DSP system at this higher frequency. However, that may not be feasible due to resource/performance demands in other parts of the system, or the data sampling rate may be fixed for specific reasons, or it may be overkill just to achieve sinc compensation.

A better solution may be to use an interpolating or 'oversampling' DAC. Such DACs change their outputs at a faster rate than the sampled data, achieving the same shift in DAC frequency response as running the whole system at the faster rate.

Interpolation means finding the values of datapoints that we do not have in between the ones we do have. If the DAC is able to calculate new sample points, it can increase its sampling rate with the advantages just discussed.

For example, if it can determine the value the signal has exactly halfway between the existing samples, the data rate can be doubled. In general, an interpolating DAC will increase the sampling rate by an integer multiple *K*, by inserting ($K - 1$) samples between each consecutive pair of existing samples.

Multiples of 2, 4 and 8 are common, and interpolating DACs often have selectable values of *K*.

A block diagram showing the structure of an interpolating DAC is shown in Fig.6, while some example waveforms (with $K = 2$) are shown in Fig.7.

They operate by first inserting zero samples into the data between the existing data points. This produces a set of samples at the higher Kf_s sampling rate, as shown in the middle waveform in Fig.7. To obtain the original waveform at the higher sampling rate, the zero-inserted sampled waveform is passed through a digital low-pass filter.

Sinc correction filter example

We will now look at an example analog (post-equalisation) filter. Although a digital implementation may have more benefits, the analog approach is likely to

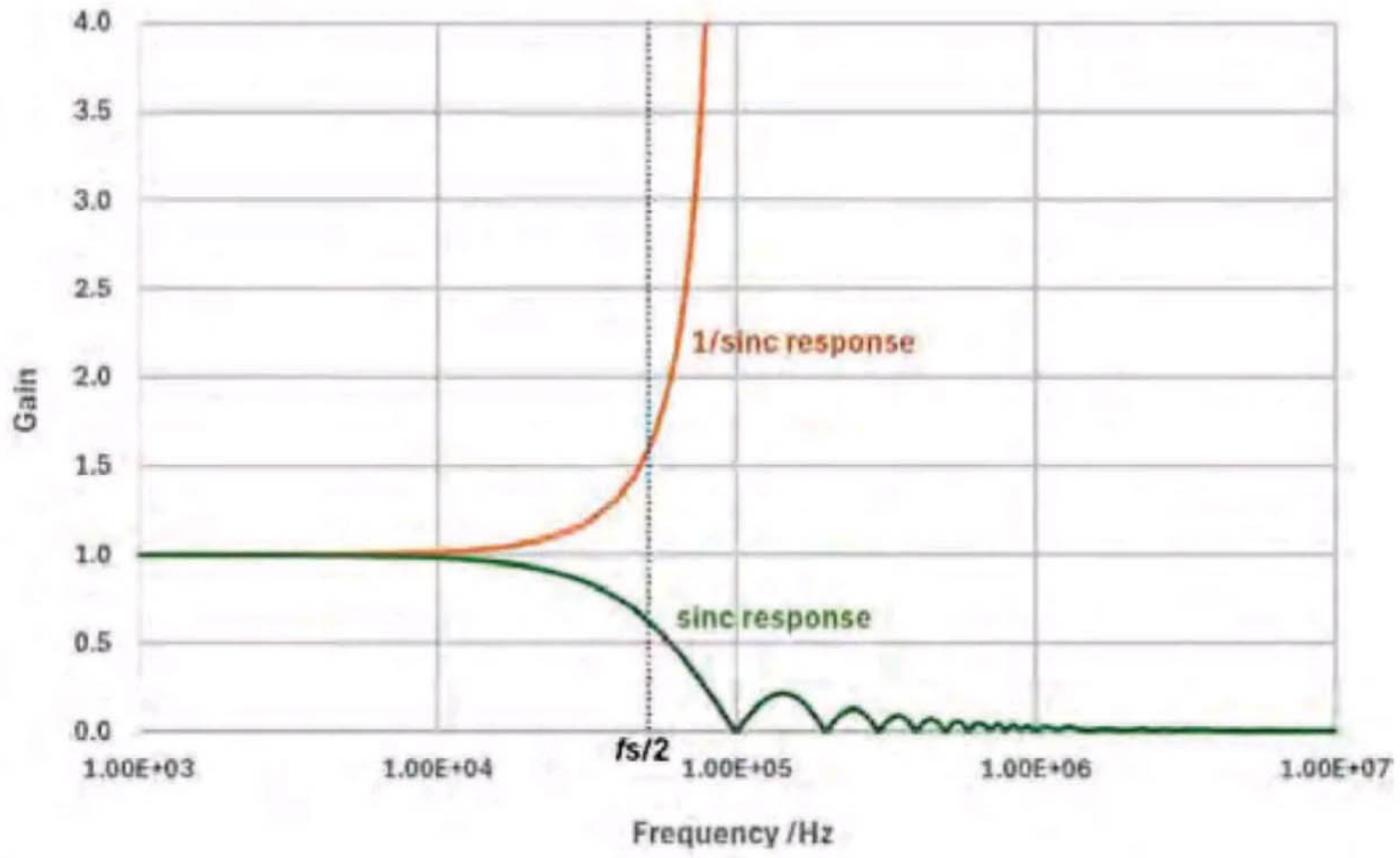


Fig.8: a DAC equalisation filter needs a 1/sinc response to $f_s/2$.

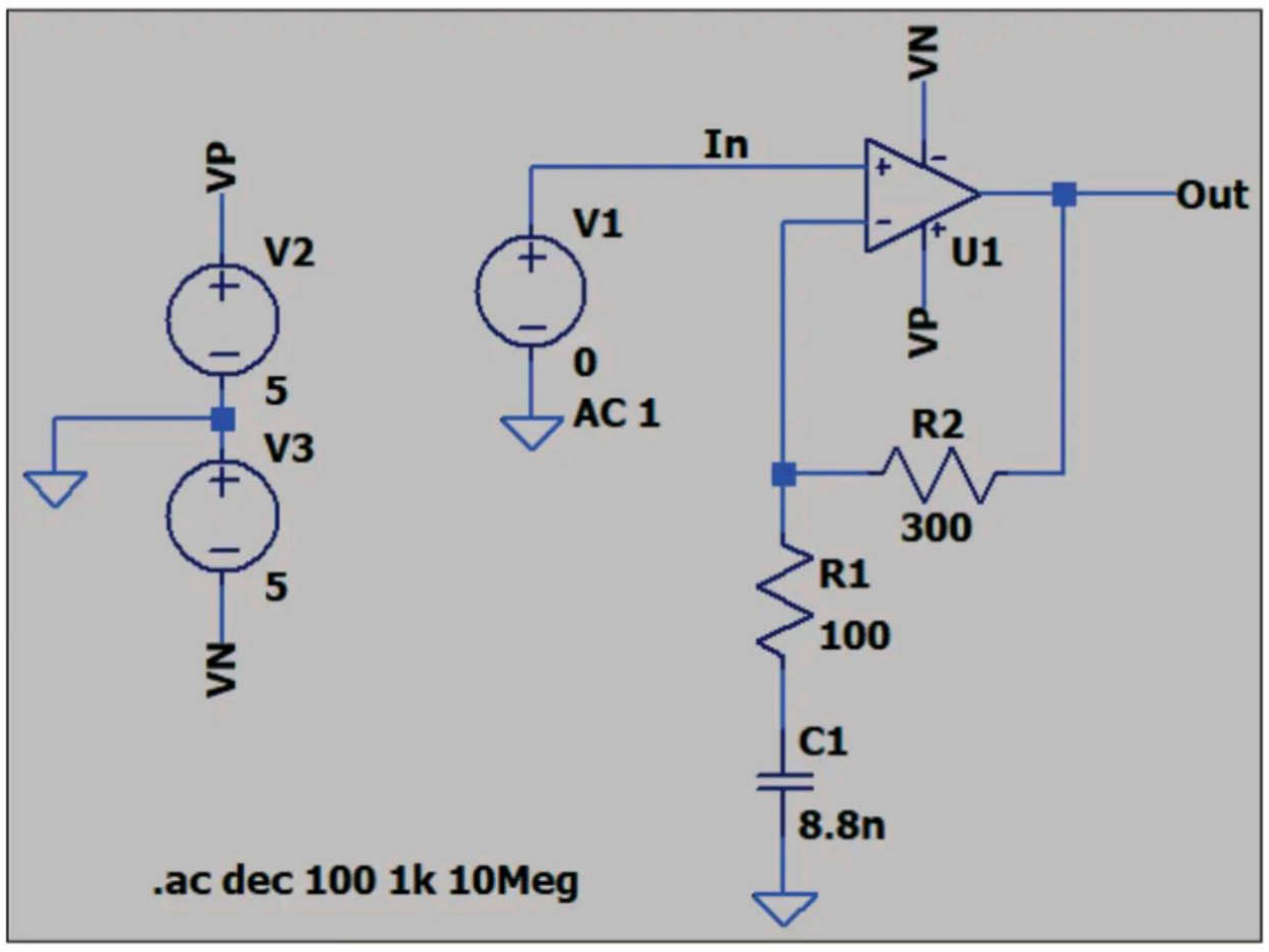


Fig.9: an LTspice high-pass shelving filter for sinc equalisation.

be more familiar (we have not discussed digital filtering yet). Readers may also find the circuit useful in other contexts; for example, similar circuits are used in audio equalisation.

We will consider a DAC with a sampling frequency of f_s = 100kHz. Fig.8 shows a plot of the DAC sinc-shaped frequency response [absolute value of sinc($\pi f/f_s$)] and the reciprocal of this value, which are labelled as “sinc response” and “1/sinc response”, respectively.

The graph was plotted using Microsoft Excel, but other spreadsheets and mathematical apps could achieve the same thing.

There is possible confusion over the units used in the sine function to calculate the sinc functions. They should be radians (the default in Excel), but the units for f and f_s can be Hertz or radians per second, because the calculation uses a ratio (so unit scaling cancels). They must be in the same units, of course!

We need to create a filter with a gain that matches the 1/sinc response, but there are a couple of points to consider before we decide exactly on how to do that. Firstly, the wanted signal only extends to at most $f_s/2$, and usually has a lower maximum, as a real system will not be processing signals right up to the Nyquist limit.

Secondly, the 1/sinc function becomes very large as we approach f_s (it goes to infinity at f_s); such large gains are neither practical nor desirable.

If the gain becomes too large, we risk clipping the signal (if present at that frequency). There’s also the risk of providing high-gain amplification of any high-frequency noise from the previous stage. Fortunately, we do not need to match the 1/sinc function in this range.

Equalisation filter circuit

The equalisation filter therefore requires a high-pass function, with a gain of 1 at low frequencies to provide the flat response there. At higher frequencies, the gain must increase but level off at some point rather than constantly increasing. This is referred to as a high-pass shelving filter. A simple analog implementation, which can be used as a post-equalisation filter, is shown in Fig.9 (an LTspice schematic).

This circuit includes an idealised op amp using the built-in UniveralOpAmp1 model with parameters (right click to set) `Avol=10G`, `GBW=100G`, `Rin=100G` and all others at zero. The supplies are not required for this op amp model, but would be needed if a real device model (or the more advanced UniveralOpAmp version) was substituted.

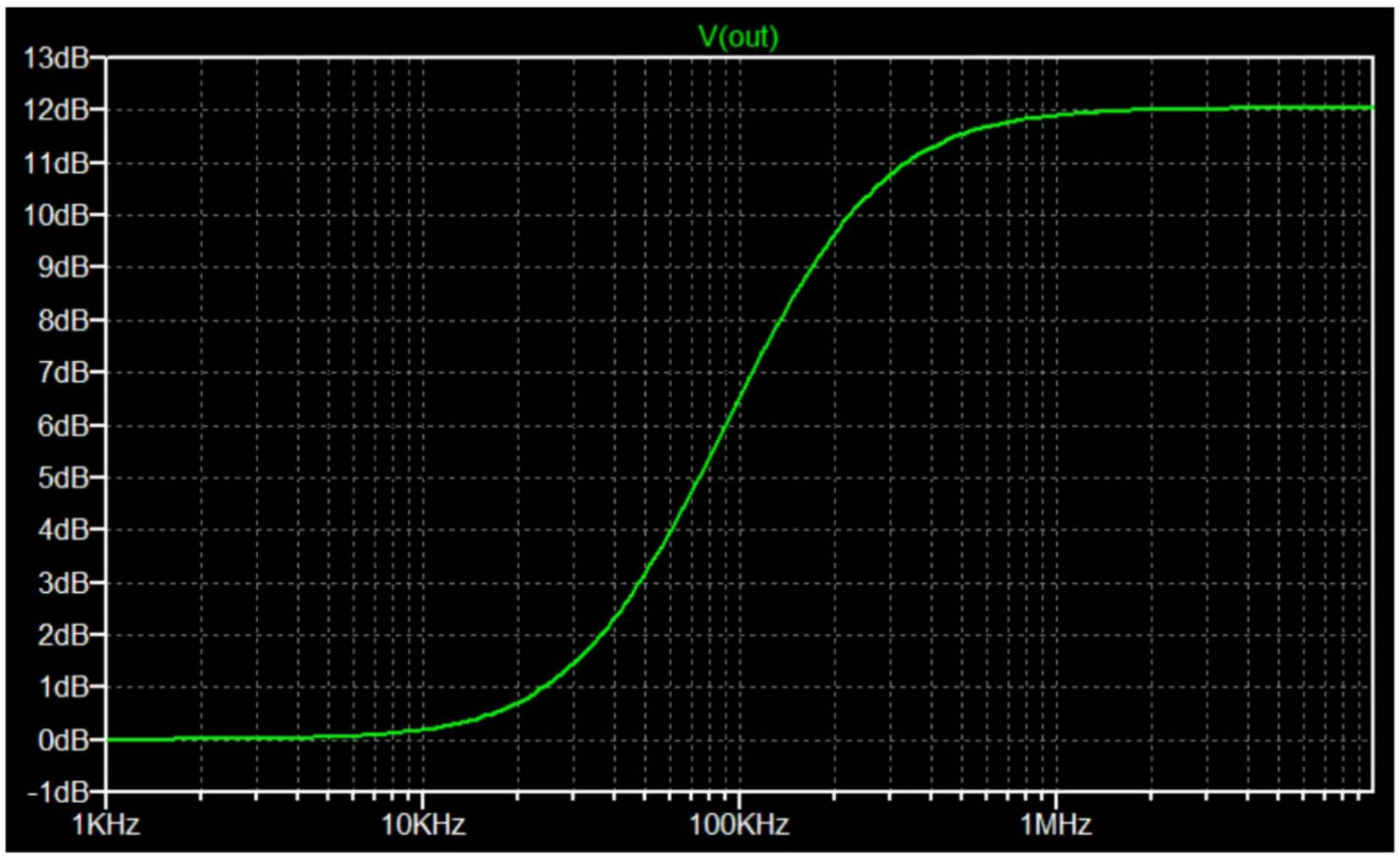


Fig.10: the frequency response of the circuit in Fig.9.

Fig.10 shows the frequency response of the circuit of Fig.9. The gain is unity (0dB) at frequencies below about 10kHz and increases as the frequency increases, levelling out at four times (12dB) above about 1MHz. This levelling off is the required shelving behaviour.

To implement the specific 1/sinc response of Fig.8, we have to match the response of the circuit to the sinc function below approximately 50kHz. To do this, we need to some formulae to help calculate circuit values.

The Fig.9 circuit is like a non-inverting op amp amplifier (see Fig.11) in which the gain-setting components are frequency-dependent – hence the filtering action. The gain of a basic non-inverting op amp amplifier is given by the well-known formula:

$$1+\frac{R_F}{R_G}$$

R_F is the feedback resistor value (the one between the output and inverting input), while R_G is the grounded resistor value (from the inverting input to ground). This is because the gain of the circuit is equal to 1/β, where β is the fraction of the output signal fed back to the inverting input.

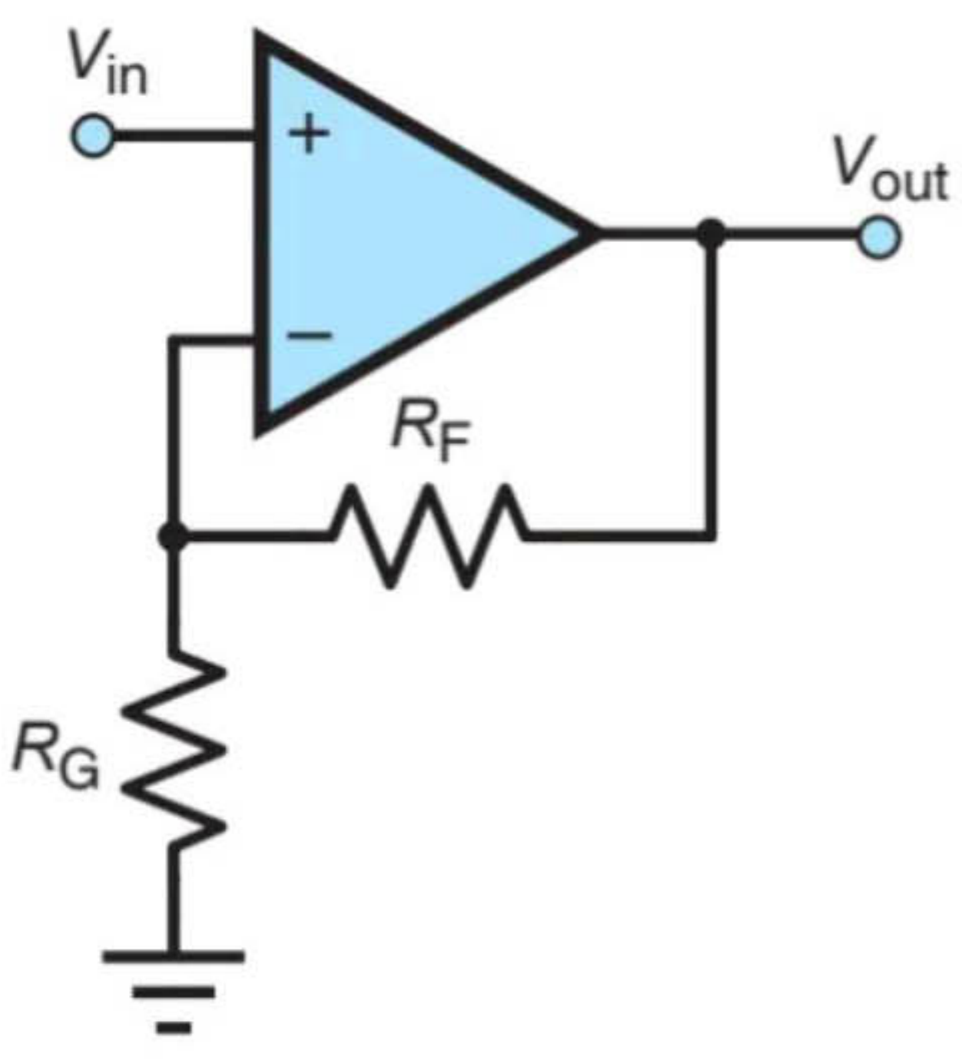


Fig.11: a basic non-inverting amplifier made from an op amp and two resistors.

The feedback is provided by the potential divider formed by R_F and R_G, and the well-known potential divider formula gives the value of β as the lower resistor (R_G) divided by the total resistance ($R_F + R_G$), that is:

$$\beta = \frac{R_G}{R_G + R_F}$$

Taking 1/β gives the gain formula.

Filter formulae

For the filter, the lower divider resistor is replaced by the series combination of R_1 and C_1. This is an impedance (call it Z_1) rather than a simple resistance.

A full analysis can be done using Laplace transform circuit values, but as we only need to calculate the magnitude of the impedance (not the phase), we can use the magnitude of Z_1 in place of lower resistor in the formula for β. This is given by:

$$\sqrt{R_1^2 + X_{C1}^2}$$

Here, X_{C1} is the reactance (effective resistance) of C_1. Note that we have to square, sum and root these two values – they cannot just be added, like we could do for two resistors in series. The reactance of a capacitor of value C is $X_C = 1 \div (2\pi fC)$.

Similarly, the total effective resistance to use in the potential divider formula for β is the root-square sum of the combined resistors ($R_1 + R_2$) and X_{C1}, that is:

$$\sqrt{(R_1 + R_2)^2 + X_{C1}^2}$$

So, the gain of the filter in Fig.7 (1/β) is:

$$\frac{\sqrt{(R_1 + R_2)^2 + X_{C1}^2}}{\sqrt{R_1^2 + X_{C1}^2}}$$

This is unfortunately a somewhat cumbersome formula.

At very low frequencies, the value of X_{C1} is very large (a capacitor's impedance increases as frequency decreases). The large value of X_{C1} means that the resistors have very little effect on the total value inside the square roots, so they can be ignored. Removing them causes the two X_{C1} terms to cancel. So, the gain at low frequencies is unity (0dB), as seen in Fig.10.

At very high frequencies, the value of X_{C1} is very small – it tends to zero, so we can remove it from the equation, and the gain becomes the same as the inverting amplifier, with R_1 as R_F and R_2 as R_G, that is:

$$1 + \frac{R_2}{R_1}$$

For the circuit in Fig.9, this is 1 + 300/100 = 4, or 12dB as seen in Fig.10. In general, we can choose two resistors to set the higher gain value. The capacitor then determines the frequency at which the gain starts to increase.

To design the filter, we could set the lower cut-off frequency to match a point on the 1/sinc response curve. 'Cutoff frequency' commonly refers to a 3dB change in gain, but this is by definition rather than a fundamental rule; some filters use a different definition.

The shelving filter can have two cutoff frequencies defined: a lower one (f_L), where the gain first rises a certain amount (eg, by 3dB above 0dB), and an upper one (f_U), a certain amount below the high frequency level-off (eg, 3dB below 12dB).

Shelving filters may have less than 6dB difference between the two level gains, so 3dB cutoff definition may not be appropriate; however, in this case, it is fine. Furthermore, the sinc frequency response for a 100kHz DAC is down by 3dB at about 44.2kHz, which is a good portion of the Nyquist frequency and may be a suitable starting point to try to match the 1/sinc and filter gains.

+3dB corresponds to a gain of $\sqrt{2}$. If we equate the filter gain expression above to $\sqrt{2}$, we can solve for f to find the lower

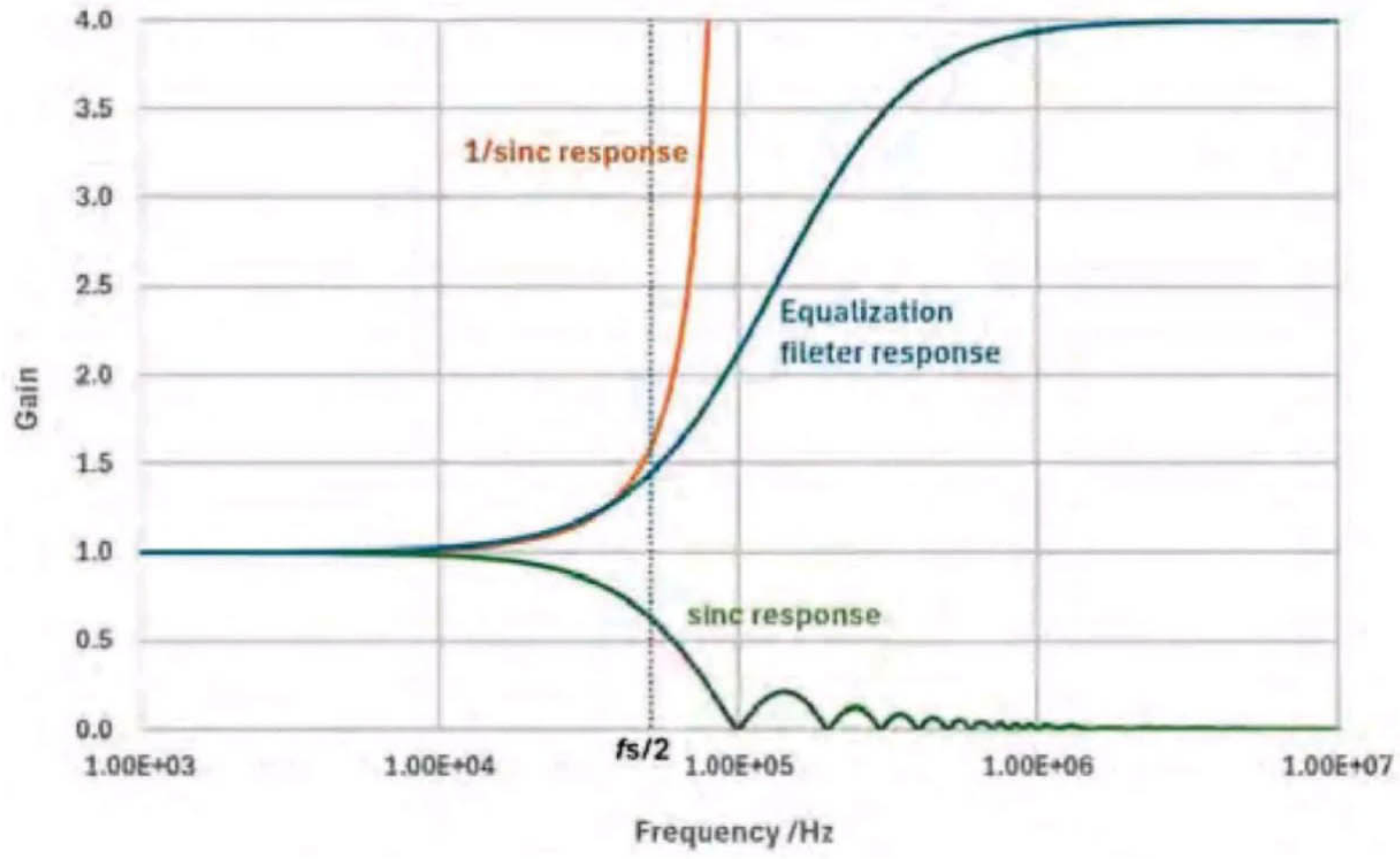


Fig.12: the equalisation filter response compared with 1/sinc.

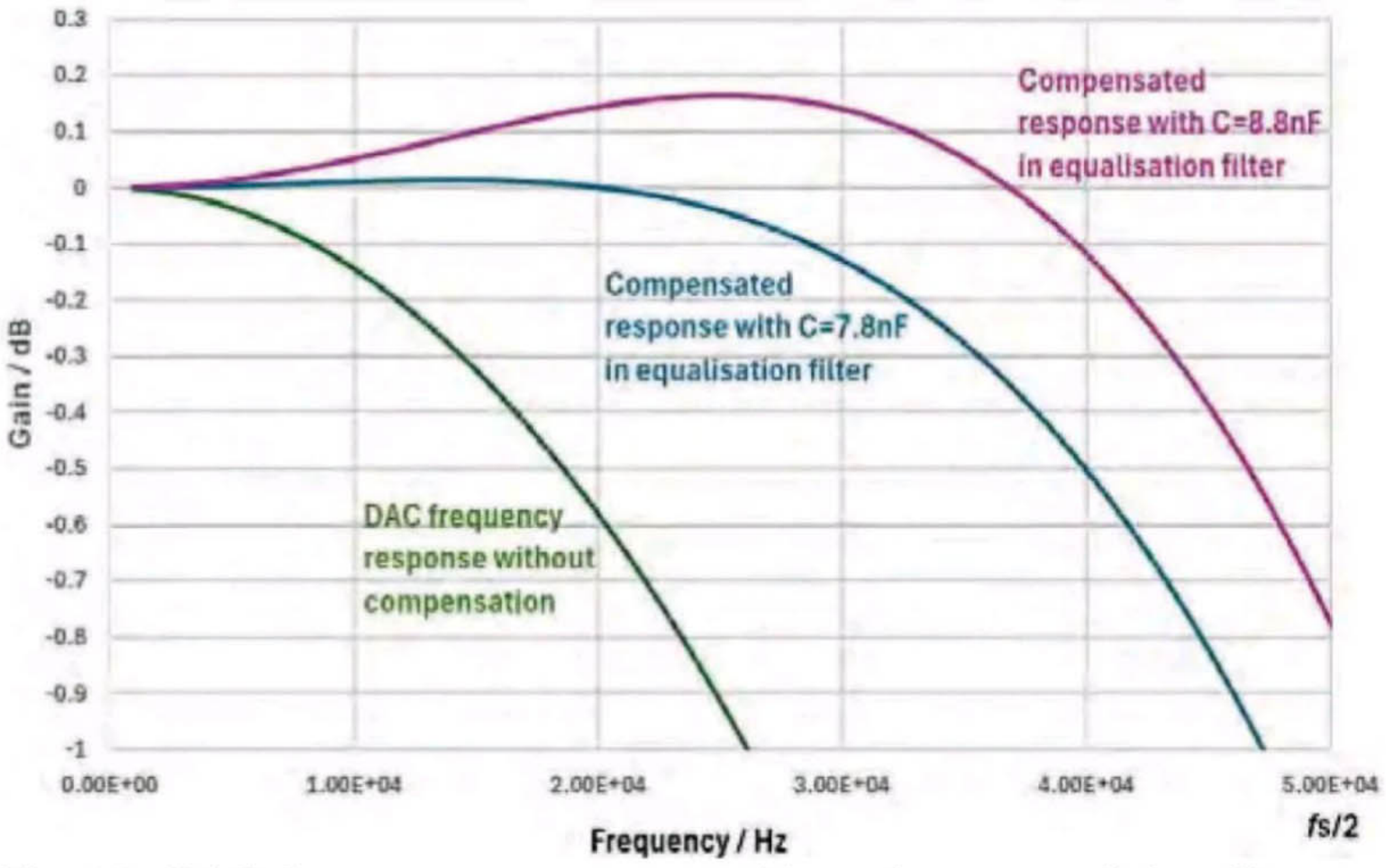


Fig.13: DAC frequency response with various equalising filters.

HAMMOND®

1557 Polycarbonate IP68

Learn more:
hammondmfg.com/1557

uksales@hammfg.com • 01256 812812

3dB cutoff frequency. After a bit of algebra, we get:

$$f_L = \sqrt{\frac{1}{(2\pi C_1)^2 (2R_1R_2 + R_2^2 - R_1^2)}}$$

Conveniently, this equation allows us to swap f_L and C_1, so we can find a capacitor for a required cutoff (assuming we have already chosen the values of R1 and R2 to set the upper gain).

Using the resistor values from Fig.7 and f_L = 44.2kHz gives C_1 = 9.6nF. This is close to the value in Fig.7, but as we will see shortly, slightly different values may prove more optimal. However, this gives us the something in the right range.

Other values we can use to characterise the key frequencies in a filter's response are the poles and zeros. These have a specific definition related to Laplace domain analysis of the frequency response, and determine the 'corner' frequencies at which the slope of the gain magnitude changes.

After a zero, the slope increases by 20dB per decade, and after a pole, it decreases by 20dB per decade. For the filter in Fig.7, there is a zero at:

$$\frac{1}{2\pi C_1 R_1}$$

This is the lower frequency corner which causes the gain slope to increase. There is a pole at:

$$\frac{1}{2\pi C_1 (R_1 + R_2)}$$

This is the higher frequency corner which causes the gain slope to decrease and hence level off.

Matching the 1/sinc response

Fig.12 shows the equalisation filter response (with the values as in Fig.7) added to the sinc function plots. This was obtained by calculating the filter gain equation given above in Excel.

This confirms a close match to the required 1/sinc up to near the Nyquist frequency, so basically it fulfils our requirements. Once this is set up in Excel, it is easy to tweak the capacitor or resistor values to see how they influence the matching of the responses.

The purpose of the equalisation filter is to compensate for the attenuation caused by the DAC's frequency response. Fig.13 shows the response of the 100kHz DAC considered in this example together with the combined response of the DAC and equalisation filter from Fig.9, with C_1 = 8.8nF and C_1 = 7.8nF. Again, the plots were produced in Excel, with the combined responses being obtained by multiplying the DAC's sinc function and filter gain at each frequency.

Simulation files

**Most months, LTSpice is used to support descriptions and analysis in Circuit Surgery.
The examples and files are available for download from the PE website:
https://bit.ly/pe-downloads**

This is plotted in dB by calculating $20\log_{10}(G)$ for each gain value G in Excel.

In both cases, with the filter added, the attenuation at higher frequencies is reduced, showing that the equalisation filter is doing its job. The gain of the DAC-plus-filter remains within a fraction of a decibel of 0dB to frequencies much closer to the Nyquist frequency than with just the DAC.

Fig.13 also illustrates that that we can have different shapes for the combined DAC and equalisation filter response. With C_1 = 7.8nF, we get a very flat response, within ±0.015dB of 0dB to beyond 20kHz. With C_1 = 8.8nF, the response remains within ±0.2dB to beyond 40kHz but is less flat than the 7.8nF case.

Using C_1 = 9.6nF as calculated above to match the 3dB points give an even more peaked response. Peaked responses for a DAC-plus-filter like this may be useful because they can help to compensate for the roll-off of the reconstruction filter near its cutoff frequency. PE

Fluke/Philips PM3092 Oscilloscope
2+2 Channel 200MHz Delay TB,
Autoset etc – £250

STEWART OF READING
17A King Street, Mortimer, near Reading, RG7 3RS
Telephone: **0118 933 1111** Fax: **0118 933 2375**
USED ELECTRONIC TEST EQUIPMENT
Check website **www.stewart-of-reading.co.uk**
(ALL PRICES PLUS CARRIAGE & VAT)
Please check availability before ordering or calling in

HP 54600B Oscilloscope
Analogue/Digital Dual Trace 100MHz
Only £75, with accessories £125

LAMBDA GENESYS	PSU GEN100-15 100V 15A Boxed As New	£400
LAMBDA GENESYS	PSU GEN50-30 50V 30A	£400
IFR 2025	Signal Generator 9kHz – 2.51GHz Opt 04/11	£900
IFR 2948B	Communication Service Monitor Opts 03/25 Avionics	POA
IFR 6843	Microwave Systems Analyser 10MHz – 20GHz	POA
R&S APN62	Syn Function Generator 1Hz – 260kHz	£295
Agilent 8712ET	RF Network Analyser 300kHz – 1300MHz	POA
HP8903A/B	Audio Analyser	£750 – £950
HP8757D	Scaler Network Analyser	POA
HP3325A	Synthesised Function Generator	£195
HP3561A	Dynamic Signal Analyser	£650
HP6032A	PSU 0-60V 0-50A 1000W	£750
HP6622A	PSU 0-20V 4A Twice or 0-50V 2A Twice	£350
HP6624A	PSU 4 Outputs	£400
HP6632B	PSU 0-20V 0-5A	£195
HP6644A	PSU 0-60V 3.5A	£400
HP6654A	PSU 0-60V 0-9A	£500
HP8341A	Synthesised Sweep Generator 10MHz – 20GHz	£2,000
HP83630A	Synthesised Sweeper 10MHz – 26.5 GHz	POA
HP83624A	Synthesised Sweeper 2 – 20GHz	POA
HP8484A	Power Sensor 0.01-18GHz 3nW-10µW	£75
HP8560E	Spectrum Analyser Synthesised 30Hz – 2.9GHz	£1,750
HP8563A	Spectrum Analyser Synthesised 9kHz – 22GHz	£2,250
HP8566B	Spectrum Analsyer 100Hz – 22GHz	£1,200
HP8662A	RF Generator 10kHz – 1280MHz	£750
Marconi 2022E	Synthesised AM/FM Signal Generator 10kHz – 1.01GHz	£325
Marconi 2024	Synthesised Signal Generator 9kHz – 2.4GHz	£800
Marconi 2030	Synthesised Signal Generator 10kHz – 1.35GHz	£750
Marconi 2023A	Signal Generator 9kHz – 1.2GHz	£700

Marconi 2305	Modulation Meter	£250
Marconi 2440	Counter 20GHz	£295
Marconi 2945/A/B	Communications Test Set Various Options	POA
Marconi 2955	Radio Communications Test Set	£595
Marconi 2955A	Radio Communications Test Set	£725
Marconi 2955B	Radio Communications Test Set	£800
Marconi 6200	Microwave Test Set	£1,500
Marconi 6200A	Microwave Test Set 10MHz – 20GHz	£1,950
Marconi 6200B	Microwave Test Set	£2,300
Marconi 6960B	Power Meter with 6910 sensor	£295
Tektronix TDS3052B	Oscilloscope 500MHz 2.5GS/s	£1,250
Tektronix TDS3032	Oscilloscope 300MHz 2.5GS/s	£995
Tektronix TDS3012	Oscilloscope 2 Channel 100MHz 1.25GS/s	£450
Tektronix 2430A	Oscilloscope Dual Trace 150MHz 100MS/s	£350
Tektronix 2465B	Oscilloscope 4 Channel 400MHz	£600
Farnell AP60/50	PSU 0-60V 0-50A 1kW Switch Mode	£300
Farnell XA35/2T	PSU 0-35V 0-2A Twice Digital	£75
Farnell AP100-90	Power Supply 100V 90A	£900
Farnell LF1	Sine/Sq Oscillator 10Hz – 1MHz	£45
Racal 1991	Counter/Timer 160MHz 9 Digit	£150
Racal 2101	Counter 20GHz LED	£295
Racal 9300	True RMS Millivoltmeter 5Hz – 20MHz etc	£45
Racal 9300B	As 9300	£75
Solartron 7150/PLUS	6½ Digit DMM True RMS IEEE	£65/£75
Solatron 1253	Gain Phase Analyser 1mHz – 20kHz	£600
Solartron SI 1255	HF Frequency Response Analyser	POA
Tasakago TM035-2	PSU 0-35V 0-2A 2 Meters	£30
Thurlby PL320QMD	PSU 0-30V 0-2A Twice	£160 – £200
Thurlby TG210	Function Generator 0.002-2MHz TTL etc Kenwood Badged	£65

HP/Agilent HP 34401A Digital
Multimeter 6½ Digit £325 – £375

HP33120A	Function Generator 100 microHz – 15MHz	£350
HP53131A	Universal Counter 3GHz Boxed unused	£600
HP53131A	Universal Counter 225MHz	£350
Audio Precision	SYS2712 Audio Analyser – in original box	POA
Datron 4708	Autocal Multifunction Standard	POA
Druck DPI 515	Pressure Calibrator/Controller	£400
Datron 1081	Autocal Standards Multimeter	POA
ENI 325LA	RF Power Amplifier 250kHz – 150MHz 25W 50dB	POA
Keithley 228	Voltage/Current Source	POA
Time 9818	DC Current & Voltage Calibrator	POA

Marconi 2955B Radio
Communications Test Set – £800

The MG4 XPower

Electric Car

by Julian Edgar

No technological change seems to inspire love/hate emotions like electric vehicles (EVs). Many people are either intensely for them or intensely against. The truth is much more nuanced, as Julian Edgar describes after nine months and 20,000km with his MG4 XPower EV.

Having been interested in car tech for over 40 years, I've watched the advent of EVs with fascination. I first drove a Tesla 15 years ago and was enormously impressed. However, especially living in a rural area, I couldn't see the worth of buying an EV until about nine months ago.

Then, an EV was released that, for the first time in the modern history of electric vehicles, had a significant advantage over any new internal combustion engine (ICE) car in existence. That advantage was the price for the level of performance!

With the release of the Chinese-made MG4 XPower, extraordinary performance became available at a cost that, in round terms, was about half that of an equivalent ICE car. For £37,190, you can now get performance that is the province of ICE cars costing at least £70,000. That is simply incredible; it is the most significant change in cars I have ever seen.

Of course, if the car itself were terrible, that apparent advantage would count for nought. I went to a dealer and drove the MG4 XPower and was very impressed, so I bought it. Now, nine months later, what do I think of the MG4 – and of owning an EV, generally?

The MG4 XPower

The venerable UK brand MG has been owned by Chinese company SAIC Motor since 2007 (although it was initially acquired from BMW by another Chinese company in 2005). While the company maintains a small UK design base, perhaps 95% of the car is designed and manufactured in China.

A mid-sized hatchback (some people say the car is small; it could only be termed that in an era when very large cars have been normalised), most models of the MG4 use a rear-mounted 150kW (200bhp) electric motor and a 64kWh 400V lithium-ion battery pack. That under-floor battery weighs 409kg.

The sportier XPower uses a 170kW (230bhp) rear electric motor and a 150kW (200bhp) front electric motor, both of which are three-phase, permanent magnet synchronous designs. Compared to the standard car, the XPower has larger brakes, revised suspension and different interior and exterior trim. Its claimed 0-100km/h (0-62mph) time is just 3.8 seconds.

That is phenomenally fast – as fast as a Ferrari from a few years ago. The XPower weighs 1800kg (about 4000 pounds), which is not particularly heavy in today's terms.

As opposed to a hybrid car that uses a combination of an ICE engine, HV battery and electric motor, an EV must be charged from mains power. The time that takes depends on the car itself and the charger to which it is connected. With the MG4, the DC charging power to the battery pack can be up to 140kW, meaning that a normal 10% to 80% charge takes about 30 minutes (charging speed isn't linear).

The Chinese-built MG4 is one of the new breed of cost-effective electric cars currently available. This is the XPower version, a very fast car priced about half the equivalent car with a petrol engine. The high-voltage battery is mounted under the floor, with clever styling disguising the increased height of the lower edge of the doors.

Of course, that's only when using a high-power charger such as those found at highway rest stops, shopping centres and the like. Using the provided AC charger (termed by many a 'granny' charger because it is so slow!), it takes more than 20 hours to charge the battery fully.

I use an aftermarket 3.6kW charger powered from a dedicated 15A home socket, which will charge the battery to 80% overnight from a starting level of about 20%. Electric power is limited when the battery charge drops below about 25%; as the battery charge decreases below that, the available power continues to decline.

This caused us a problem only once, when my wife was driving home with a very low battery level and had to climb a long highway hill. In that case, the car would only achieve 80km/h (50mph), which was a bit dangerous on a 110km/h (70mph) road.

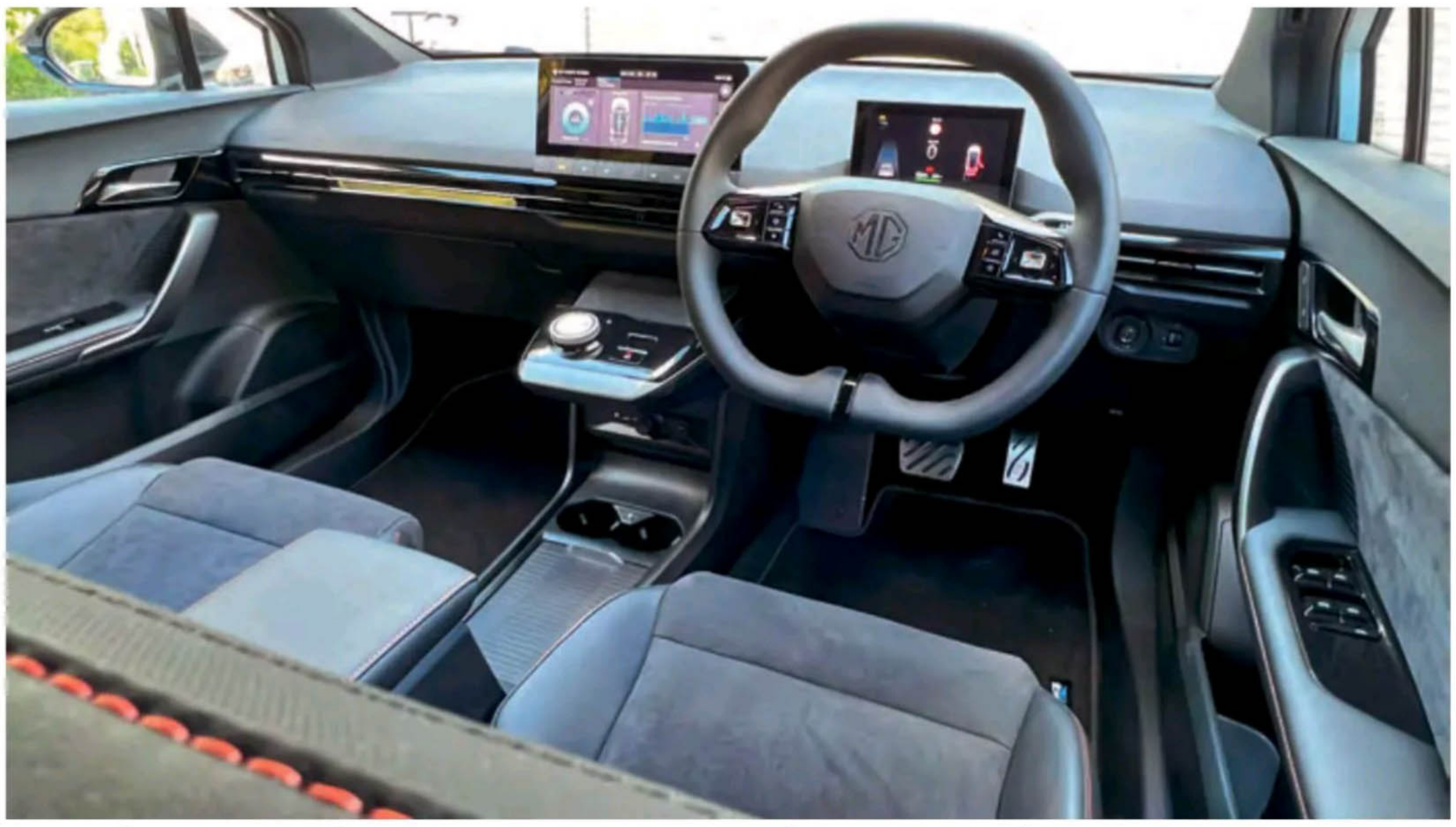
The official energy consumption of the XPower is 19kWh per 100km (30.6kWh/100mi). That has proven accurate in summer conditions, but the consumption is a bit higher in winter – nearer 20kWh per 100km (32kWh/100mi). With a 64kWh battery, and working from 80% to 10% capacity, the range is about 230km (143mi).

Why only 80% to 10%? The manufacturer suggests using the battery in that way under normal conditions and only tapping into the full capacity when undertaking long trips. Using the full battery capacity gives a range of about 330km (205mi), but doing that frequently will degrade the battery prematurely.

The displayed battery range is very accurate. Initially, I was fearful of letting the battery level get below about 15%. Judging the remaining range of ICE vehicles based on fuel levels can be hit and miss, so I thought the MG4 display might suddenly drop from 15% to zero, stranding me by the side of the road and requiring a flatbed truck to get me home!

However, I now realise there are no problems in running the battery down to, say, 5% as the change in the predicted range corresponds very well with the distance travelled.

As with all EVs, the MG4 uses regenerative braking (ie, it returns power to the battery under braking). This is achieved in two ways. The first way is as you lift the accelerator pedal, the car automatically starts to brake regeneratively, a bit like engine braking with an ICE car in gear. The amount of regeneration can be seen on the driver display; it is seamlessly varied with the right foot.



As with many modern cars, instruments and most controls are via LCD screens. The centre is a touch screen; the buttons below it are the only buttons on the car!

The environmental footprint

One reason many people are for or against EVs relates to the environmental footprint. There is so much information (and misinformation) on this topic. However, major peer-reviewed studies show that the total lifecycle environmental footprint (including building the car, running it and disposing of it) is less for an EV than an ICE car.

That is the case even when the EV is charged mainly from coal power. However, hybrid cars can be very close depending on the exact power-generating mix. But for me, some of this debate loses the wood for the trees: it's far better for the environment to ride a bicycle or take public transport. Or even to retain the old ICE car and use it only for short trips.

The second way regenerative braking occurs is when the brake pedal is pressed. That increases the level of regeneration over that achieved by lifting your foot off the accelerator pedal and, if the brake pedal is applied harder, the friction (conventional) brakes also help to slow the car.

Regenerative braking is so effective that the disc brakes become slightly rusty from a lack of use and can squeak a little when applied. One hard braking event then cleans them again.

The stand-out feature of the XPower is its amazing drivetrain. With 600Nm of torque, the XPower is extraordinarily strong, linear, refined and responsive. The only ICE car I've driven that comes close to its effortless performance is a twin-turbo V-12 Mercedes and, of course, the XPower is much faster.

We're talking about a wave of torque that just hurls the car forward, making driving situations like overtaking on country roads ridiculously easy. The drivetrain is the most impressive I have driven in 35 years of professionally testing cars; it makes my Porsche 981 Cayman engine and transmission look positively agricultural.

The ability to 'play a tune' on the accelerator pedal, seamlessly moving from immense power to braking, is simply wonderful. It's a delight I enjoy every time I get into the car, whether in city stop/start traffic or driving down a twisty country road.

The design and build quality of the MG4 are excellent. The paint is very good and panel margins (gaps between adjoining panels) are consistent. Even when delving under the plastic covers positioned over so many of the mechanicals, the engineering and build quality look good.

You must search hard to find deficiencies, but an example is the stitching on the underside of the head restraints. It looks as if the person operating the machine was looking the other way at times!

The interior of the car is quite minimalistic; some would call it plain. There are the two displays, a short row of buttons, a charging pad for your phone and not much else. To some people, it looks cheap and nasty; to others, it is sleek and modern. I fall midway between the two camps – I'd like to see more control buttons and bigger screens, but otherwise, the interior austerity doesn't concern me.

Regarding the screens, the central unit measures 10.25 inches (26cm), but unfortunately, the screen behind the steering wheel is only 7 inches (18cm). With the small font that's often used, the latter can be hard to read at a glance, although familiarity has improved this.

Nearly all the controls are operated through the central touch screen, with only seven physical buttons provided below it. The central screen can be slow to react, especially when the car is first started, and accessing controls that in other cars would be a simple button-push away can become a clumsy dance of fingers.

However, two of the steering buttons are programmable so, for example, some of the heater/air conditioner controls can be accessed through a steering wheel button and then adjusted via a steering wheel toggle.

The air conditioner uses a high-voltage electric motor to power the compressor and it works extremely well. Heating is by a resistance heater rather than using the air conditioning system as a heat pump. Interestingly, in some overseas markets, the MG4's heater does use the air conditioner; they must not think it ever gets cold in Australia!

The seats and steering wheel are heated; these work very effectively, and I tend to use these functions rather than the cabin heater itself.

Where the technology fails – and it utterly fails – is in some of the driver assistance systems. The Lane Keeping Assistant is the worst. It is so bad that it needs to be switched off; otherwise, it beeps and yanks on the steering wheel at every imagined driving misdemeanour. On unmarked country roads, it is positively dangerous.

Frustratingly, it cannot be permanently disabled but must be switched off every time the car is driven.

Another technology that is below par is the active (radar) cruise control. It's almost as if the system was not recalibrated for the greater performance of the XPower, as it tends to be too heavy-handed with both acceleration and braking. Certainly, any competent driver can be much smoother than cruise control – in this regard, even a 15-year-old Holden Commodore is much superior.

Other MG4 users have additionally reported autonomous braking for phantom events; however, luckily, I have not experienced that.

Hopefully, MG will release software patches to solve these problems. These require a dealer visit as no over-the-air updates are available despite the car having a 4G connection.

Editor's note: given that some vehicles have been remotely 'bricked' or had features removed after purchase, I think that is a good thing.

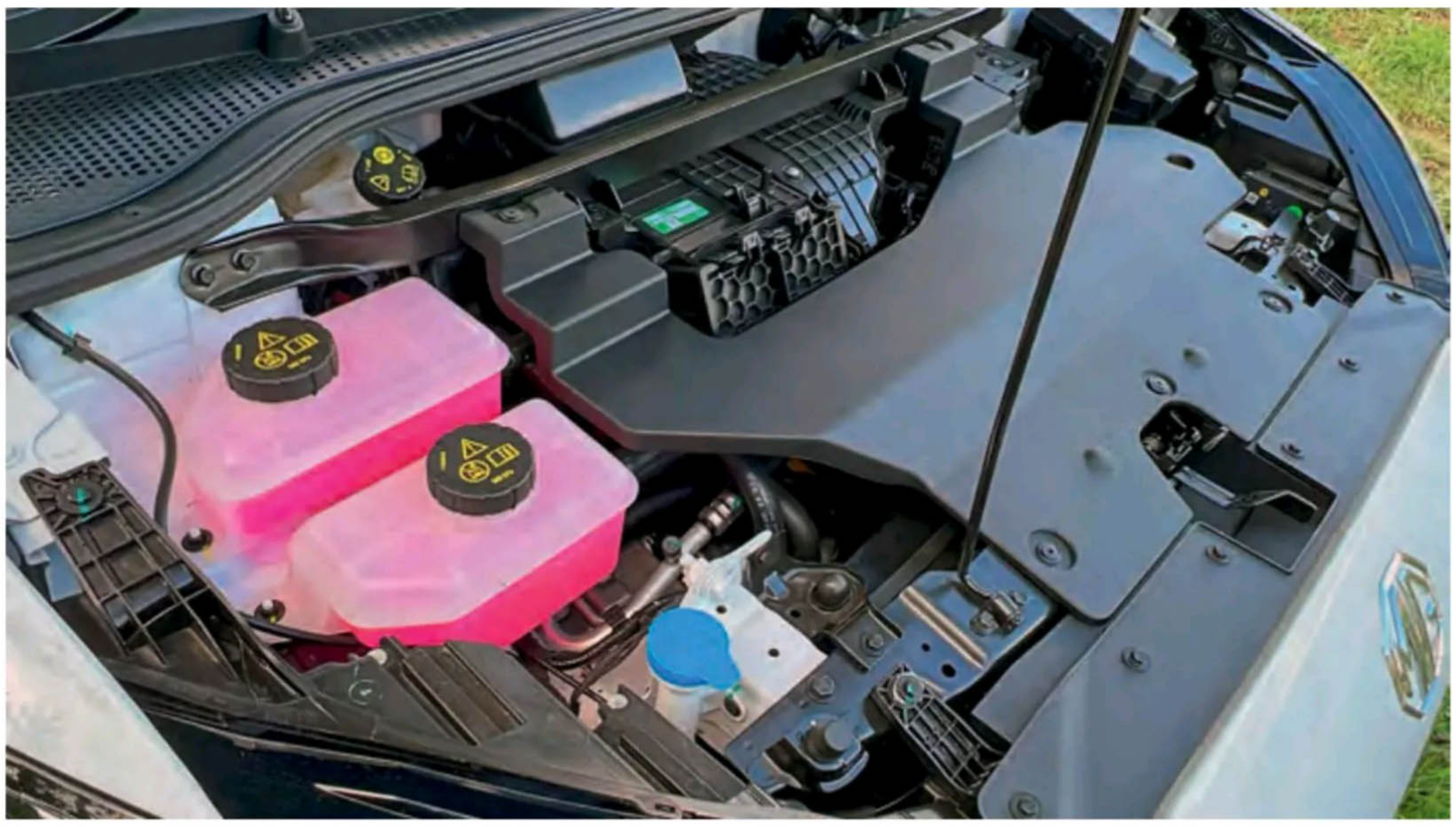
The good and the bad of EVs

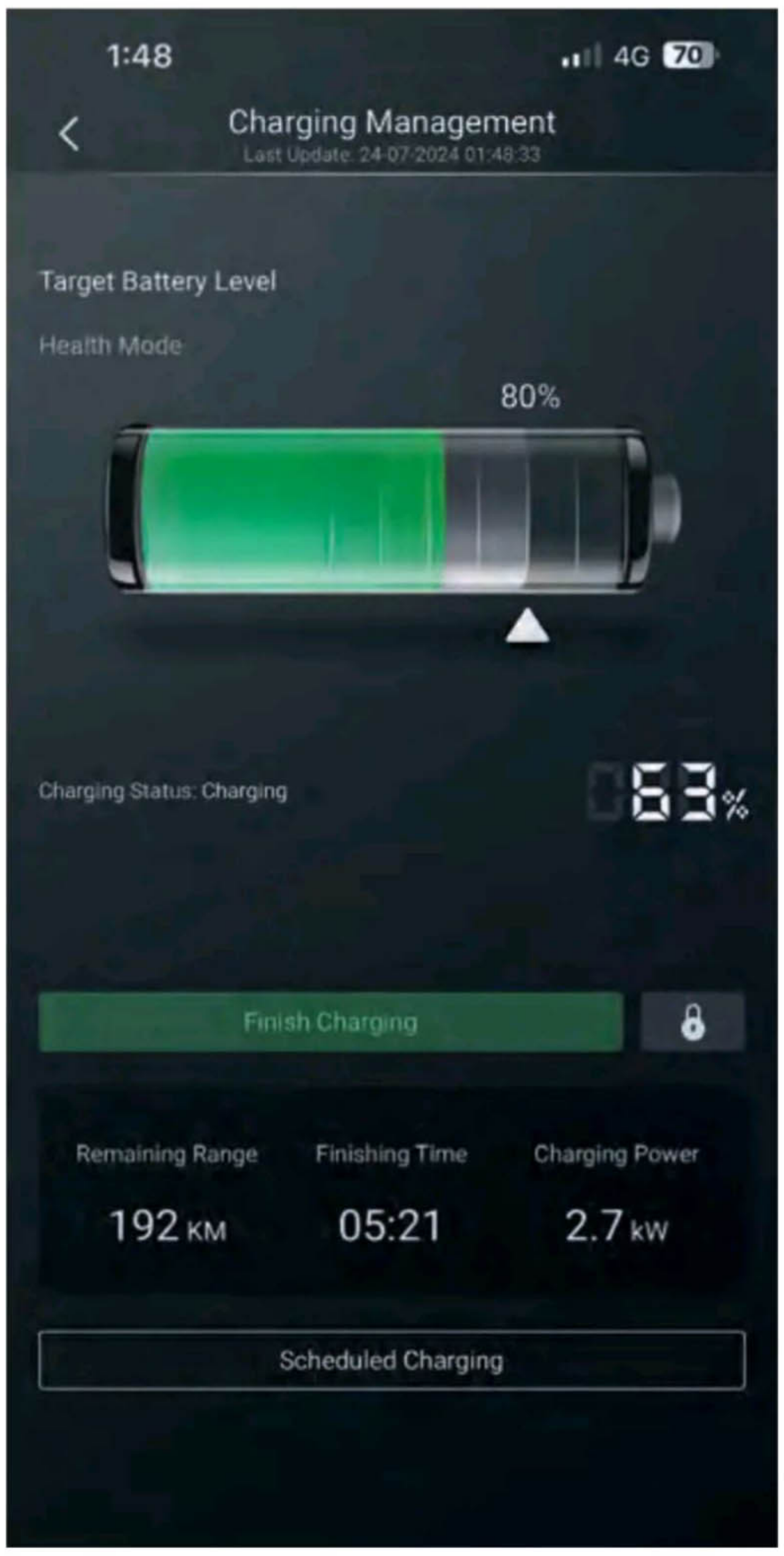
At this stage, and especially in rural and regional Australia, EVs do not make for a persuasive case for many users. More than anything else, the issues are range and charging infrastructure. Basically, for long trips, EVs are terrible.

Sure, the web is full of EV discussion groups where people claim that long trips are not only possible in EVs but are, in fact, delightful. Just stop every 2-3 hours for 30 minutes of charging, and since those stops correspond to when you'd want a break anyway, what's the problem?

The reality is different. First, you must find a high-speed charger – and compared to ICE fuel pumps, they are as rare as hens' teeth, especially off main routes. Then, the charger needs to be available. Many are broken, while others already have EVs plugged in. Imagine how long a fuel fill would take if every ICE car required half an hour at the petrol pump!

Charge Settings
Discharge Settings
Energy consumption
90% 308km
Start charging


The XPower uses both front and rear electric motors, giving all-wheel drive. This is the view under the bonnet. Its build quality is excellent overall.



Charging Management
Last Update: 24-07-2024 01:48:33

Target Battery Level

Health Mode

80%

Charging Status: Charging 63%

Finish Charging

Remaining Range	Finishing Time	Charging Power
192 KM	05:21	2.7 kW

Scheduled Charging

The MG4 has a phone app that can remotely check the battery level, lock or unlock the car and turn on the heater or air conditioner. Here, it is at 63% charge, charging at 2.7kW on its way to 80%.

The XPower sits a bit higher than a traditional hatchback due to the underfloor battery pack. It helps to keep the centre of gravity low.

Yes, you can do it, but taking an ICE car with a decent range (these days, all ICE cars) is vastly less stressful. On a long trip, the ICE car is also much quicker. Having tried it a few times, I now rarely take my MG4 on trips over 300km (186mi).

Next on the downsides is the financial uncertainty. People often quote the meagre 'fuel' cost of an EV versus an ICE car. And, especially if charged from a home PV solar system, the running costs will indeed be a lot lower. However, the major cost of buying a new car is depreciation – the amount the car loses in value each year.

At this stage, it very much looks like EVs will have fast depreciation – that has been the case in markets that are more mature than Australia in terms of EV penetration. There are several reasons why.

First, as technology rapidly improves, people value the older EVs less highly. Second, battery life. While the manufacturer often guarantees EV batteries for a set period (eg, seven years), the fine print shows that the guarantee is typically for 70% charge retention. Multiply the worst range by 0.7, and the real-world range of many EVs is likely to become marginal without any real recourse.

And what if the battery degradation is even greater than 70%? The reality of older used EVs in Australia, like the Nissan Leaf and Mitsubishi MiEV, is that these cars often have a range that's now as little as 70-80km (40-50mi). Yes, they use older battery technology – but they are real examples of older EVs. Most ICE cars still run just fine after 7-10 years (as long as they're maintained) and don't lose range.

Also, EV proponents often overlook the purchase cost. As the MG4 XPower demonstrates, in the expensive car market, EVs are now more than competitive with ICE cars. But what of those who are less wealthy? A competent second-hand ICE car can be bought for well under £6,000. No such alternatives currently exist for EVs.

As for the good aspects of EVs, they require almost no maintenance. I was initially sceptical of this, but my MG4 XPower has not seen the inside of a workshop in its first 20,000km (12,500mi). The first scheduled service interval is 40,000km (25,000mi) – for most people, that's every three years! In terms of convenience, that is a major plus.

Driven hard, I don't think the tyres on my car will last more than about 30,000km (18,500mi), so it will be a tyre shop that I first visit.

Another positive is that, depending on your use, an EV is very convenient. Plug it in each night just like your phone, and it's ready the next morning. No visits to petrol stations; just unplug and go. And, as discussed, the cost of charging an EV can be very low, especially if charging during the day from solar panels or using a low off-peak overnight tariff (where available).

I've already discussed driveability. Truly, no ICE car can compete with the superb flexibility and throttle control that EVs have. Some people suggest that EVs are rather uninvolving and aren't fun to drive – I think that is just balderdash.

So where does that leave us? I love the MG4 XPower. It's a car that is practical, a joy to drive and gives me performance unmatched by anything at its price. As for EVs in general, I think that at this stage, they're perfect for some and quite unsuitable for others. If you're relatively wealthy, live in a city, have PV panels (and especially a storage battery) and commute daily, they are perfect.

However, if you're not very wealthy, take many long trips and don't have a home charging facility with at least 3.6kW, steer clear for now.

If you're listening to people discussing EVs and they say, "EVs are fantastic!" or conversely, "EVs are terrible!", remember that they're both likely to be wrong. The truth is much more nuanced. PE

The XPower uses larger brakes and orange covers over the brake calipers. The brakes are strong but with regenerative braking, they are seldom needed!

Practical Electronics

UK readers SAVE £1 on every issue

SUBSCRIBE NOW!

01202 087631

Take out a one-year subscription and save more than £10 over the year.
Even better – save over £1 per issue if you subscribe for two years – a total saving of £26.

Overseas rates represent exceptional value

You also:
- **Avoid any cover price increase for the duration of your subscription**
- **Get your magazine delivered to your door each month**
- **Guarantee a copy, even if your local newsagent sells out**

Order by:
- **Phone or post with a cheque, postal order or credit card**
- **Or order online at: www.electronpublishing.com**

Practical Electronics

SUBSCRIPTION PRICES

Subscriptions for delivery direct to any address in:

UK: 6-months £33.99, 12-months £59.99, 24-months £114.99

Europe Airmail: 6-months £38.99, 12-months £69.99, 24-months £129.99

Rest Of The World Airmail: 6-months £46.99, 12-months £84.99, 24-months £164.99

Cheques (in **£ sterling only**) payable to *Practical Electronics* and sent to: Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Tel: 01202 087631
Email: pesubs@selectps.com

Also via our online shop at**: www.electronpublishing.com**

Subscriptions start with the next available issue. We accept MasterCard or Visa.
(For past issues see the Back Issues page.)

DIGITAL SUBSCRIPTIONS

Electronic subscriptions are available from £29.99 for 12 months, more details at: **www.electronpublishing.com**

SUBSCRIPTION ORDER FORM

☐ 6 Months: UK £33.99, Europe £38.99 (Airmail), Rest of the World £46.99 (Airmail)
☐ 1 Year: UK £59.99, Europe £69.99 (Airmail), Rest of the World £84.99 (Airmail)
☐ 2 Years: UK £114.99, Europe £129.99 (Airmail), Rest of the World £164.99 (Airmail)

To: **Practical Electronics Subscriptions**
PO Box 6337
Bournemouth BH1 9EH
United Kingdom

Tel: 01202 087631
Email: pesubs@selectps.com

I enclose payment of £ ....
(cheque/PO in £ sterling only)
payable to *Practical Electronics*

☐ Please charge my Visa/Mastercard

My card number is: ....
Please print clearly, and check that you have the number correct

Card Security Code .... Valid From Date........................
(The last 3 digits on or just under the signature strip)

Card Ex. Date ....

Name ....

Address ....

Post code .... Tel. ....

Email ....

Subscriptions start with the next available issue.

Using Electronic Modules with Jim Rowe

4-digit, 14-segment LED module

0.54" 14-segment LED HT16K33 Backpack

Instead of seven segments, this LED display module has 14 segments per character, so it can display letters, digits and even a few symbols. It has a built-in I²C serial interface, allowing popular microcontrollers like the Arduino Uno or Nano to drive it easily.

The module is about the same size as a 4-digit, 7-segment display at 50mm wide by 28mm high, with a total thickness of a little over 10mm. The two side-by-side dual-character LED displays have 14 segments per character, plus the usual decimal point LED. This allows them to reasonably display numerical digits, upper-case letters, many lower-case letters and a few symbols.

The module features an I²C serial interface that allows easy connection to just about any popular microcontroller unit (MCU). We will now look more deeply into the 14-segment LED displays, followed by the useful IC that drives them and provides the I²C interface.

The 14-segment displays

Fig.1 shows how the dual character displays used in the module have six of the seven segments used in the familiar 7-segment displays; the outer ones labelled 'a' to 'f'.

Instead of the single central horizontal segment, there are eight inner segments: three in the upper half labelled 'g', 'h' and 'j', three in the lower half labelled 'l', 'm' and 'n', and two in the centre replacing the original single horizontal segment, labelled 'p' and 'k'. This gives 14 segments in each character, not counting the decimal point.

The LEDs in these segments are connected in a common-cathode configuration, so each character (plus its decimal point LED) has a single cathode pin.

The anodes are connected to the anode of the corresponding segment of the other character, eg, segment '1a' to segment '2a' etc. That allows the segments of both displays to share pins, as shown in the internal circuit, on the right side of Fig.1.

So each dual-character display needs only 17 connection pins: 15 for the LED anodes and two for the cathodes. The displays have 18 physical pins, but one (pin 3) is not used.

Two main suppliers of these dual 14-segment displays are Kingbright (PDC54-11GWA) and Lite-On (LTP-3784E). The characters are 13.8mm (0.54in) high in both cases. These manufacturers also label the inner display segments differently, but the pin connections are the same.

The displays used in this module have segments that emit orange-yellow light, but displays with other colours are available.

Inside the HT16K33 IC

Now we can look into the IC used to drive each pair of dual 14-segment displays in the module. This is the HT16K33, made by Taiwan firm Holtek Semiconductor Inc (www.holtek.com/

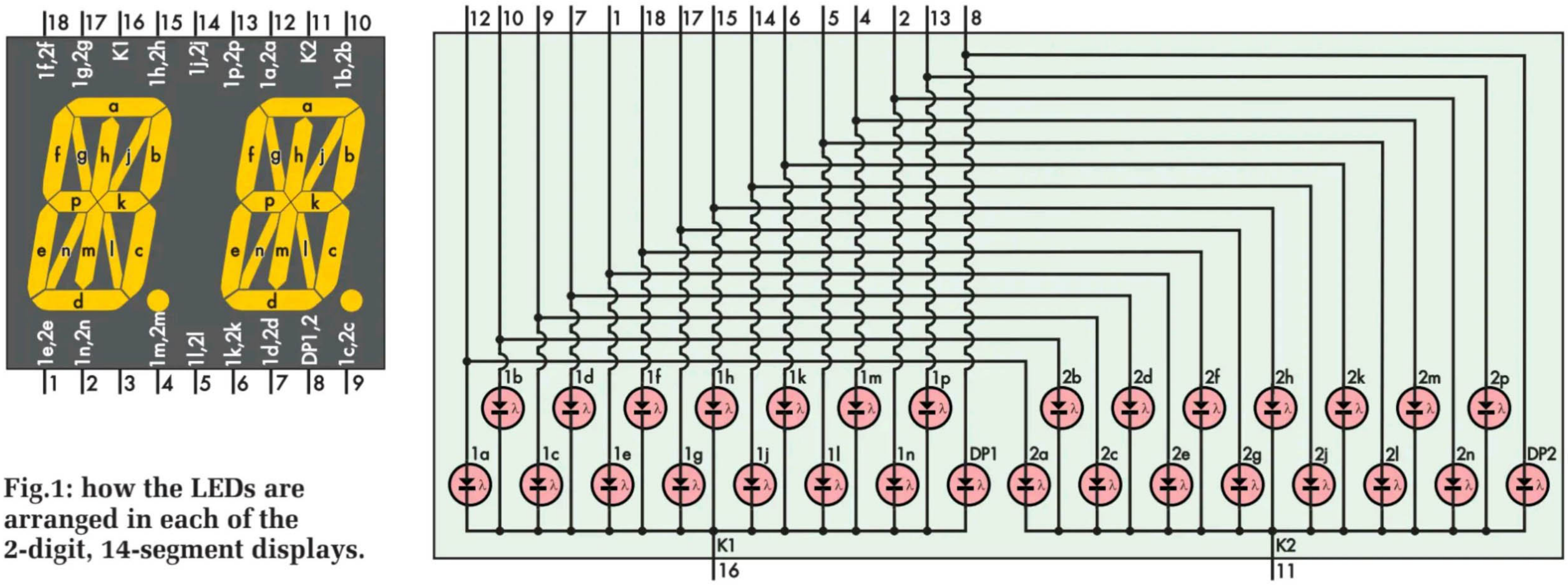


Fig.1: how the LEDs are arranged in each of the 2-digit, 14-segment displays.

page/vg/HT16K33A), which also provides the I^{2}C interface. Holtek makes a range of microcontrollers, some of which are used in popular home appliances and various other ICs, including display drivers like the HT16K33.

Holtek describes the HT16K33 as a 16×8 LED Controller Driver with RAM Mapping and an optional keypad scanning ability. It can be used for driving virtually any matrix of up to 16 × 8 LEDs, not just 14-segment alphanumeric displays, as in this module. It can also scan a matrix of 13 × 3 keys, although that feature is not used here. It can be powered from 4.5-5.5V DC.

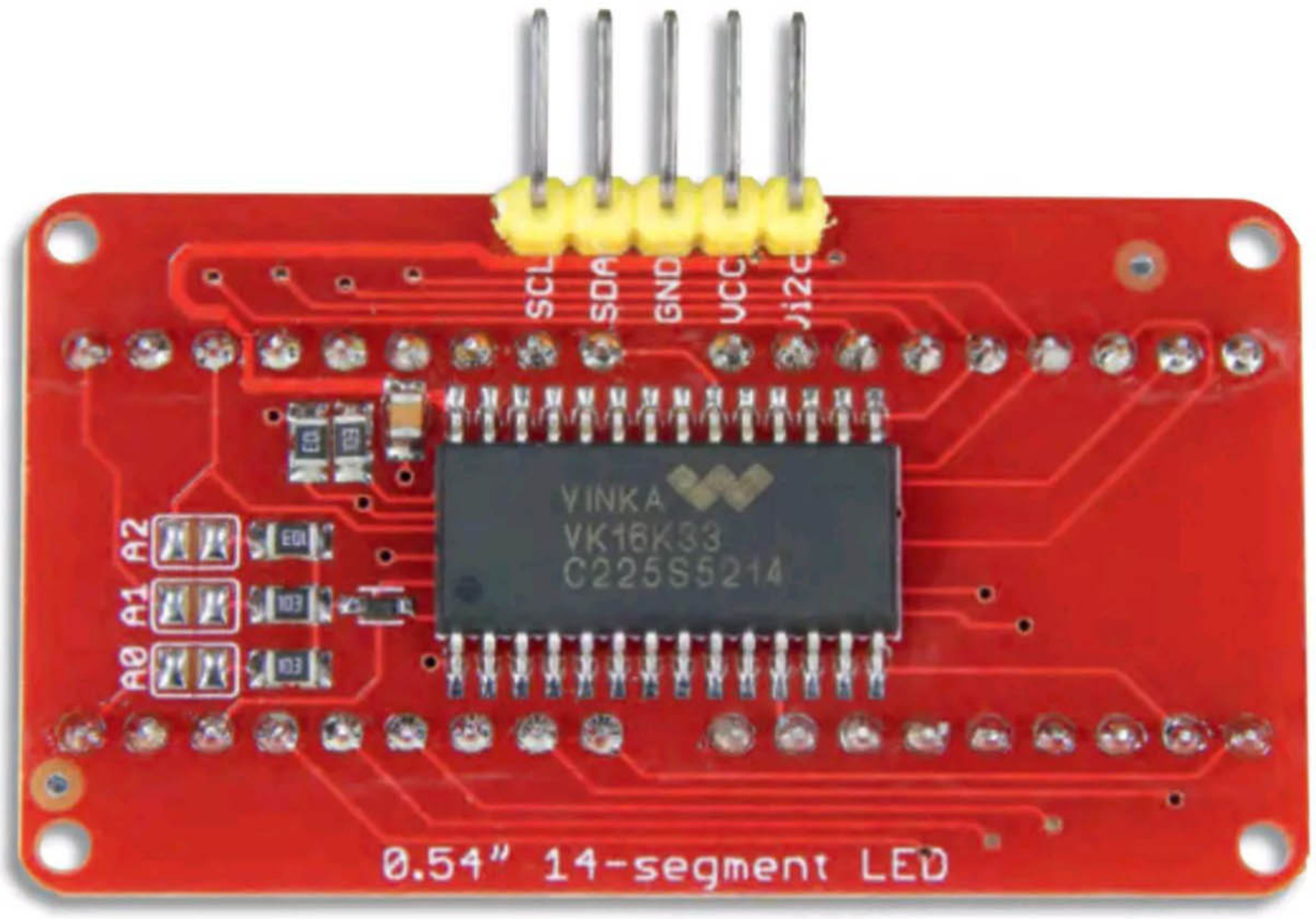
Fig.2 shows the basic block diagram of the HT16K33. The I^{2}C interface controller is at lower left, with an internal RC clock oscillator to its right feeding a timing generator, and two random-access memories (RAMs) below them. The upper RAM is for the display control data, with a capacity of 16 × 8 bits, while the lower RAM is for storing the key scanning data, if that function is used.

On the right-hand side are the two controller blocks. The upper one provides eight outputs (COM0-COM7) for control of the ‘common’ LED lines (in this case, the cathodes of the 14-segment displays) and the key scanning outputs. The COM0 output is also used to sense the desired I^{2}C address for the HT16K33, as explained shortly.

The lower controller block provides 16 outputs (ROW0-ROW15) for driving the rows of LEDs in a matrix or the segments in the 14-segment displays. It also provides inputs for sensing the desired I^{2}C address, plus inputs for the key scanning function. The power-on reset (POR) block at upper left resets most of the other blocks when power is first applied.

One of the functions of the HT16K33 not shown in Fig.2 is its ability to provide programmable 16-step dimming of the LED outputs. That is achieved by controlling the pulse width of the ROW outputs, with a range from 1/16th to 16/16th duty cycle. Another handy feature!

Finally, the HT16K33 can be programmed to have any of eight different I^{2}C addresses, from 70h to 77h, using three links on the circuit around the chip. We will see how this is done in the next section.


SCL
SDA
GND
VCC
I2C
A2
A1
A0
VINKA
VK16K33
C225S5214
0.54" 14-segment LED

The rear of the 14-segment LED module contains just a few components and the HT16K33 IC. The module’s I^{2}C address is set by the three links labelled A0-A2 on the PCB.

Note that the HT16K33 IC is now obsolete, but Holtek still sells the HT16K33A, which is pretty similar.

The module circuit

As you can see from the circuit in Fig.3, there’s not much in the module apart from the HT16K33 device itself (IC1) and the two dual 14-segment displays. Two pull-up resistors are connected between its SDA and SCL lines and the VHI input, while the HT16K33 chip is powered from the VIO input from CON1, with a 10μF capacitor providing filtering.

5-pin SIL header CON1 is used to make all the power and signal connections to the module.

Programming the module’s I^{2}C address is achieved using diode D1, three resistors and three PCB links A0-A2, shown above IC1 in Fig.3.

The anode of D1 is connected to the COM0 output (pin 2) of IC1, while its cathode connects to the three links via three 10kΩ resistors. The other ends of the links are connected to the ROW0, ROW1 and ROW2 lines of IC1, which are used as inputs when IC1 detects the desired I^{2}C address.

As shown in the small table at upper right in Fig.3, when no links are connected (A0=A1=A2=0), the module has an I^{2}C address of 70h (h = hexadecimal). If only the A0 link is connected, the address is changed to 71h; if only the A1 link is connected, this changes the address to 72h etc.

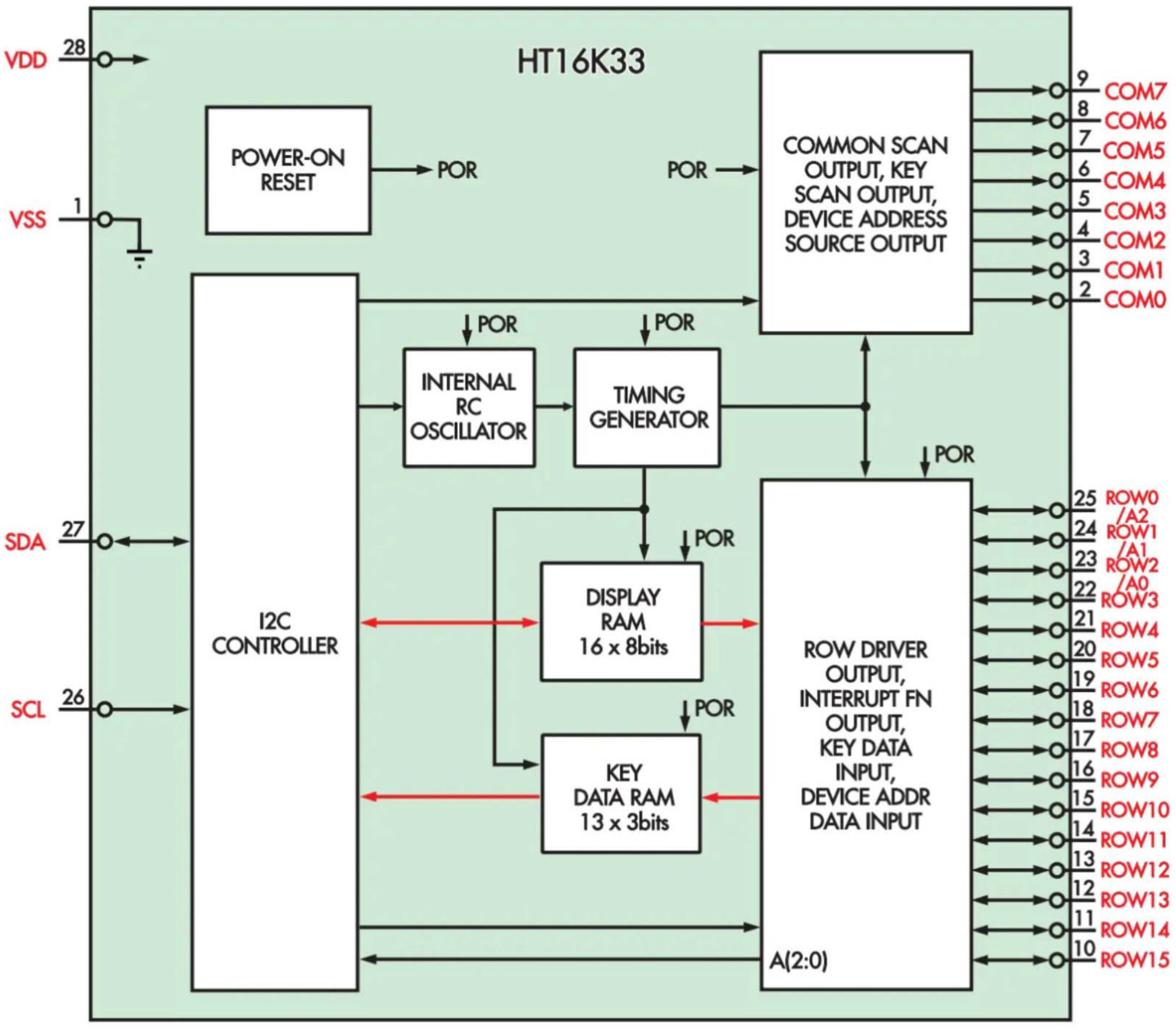
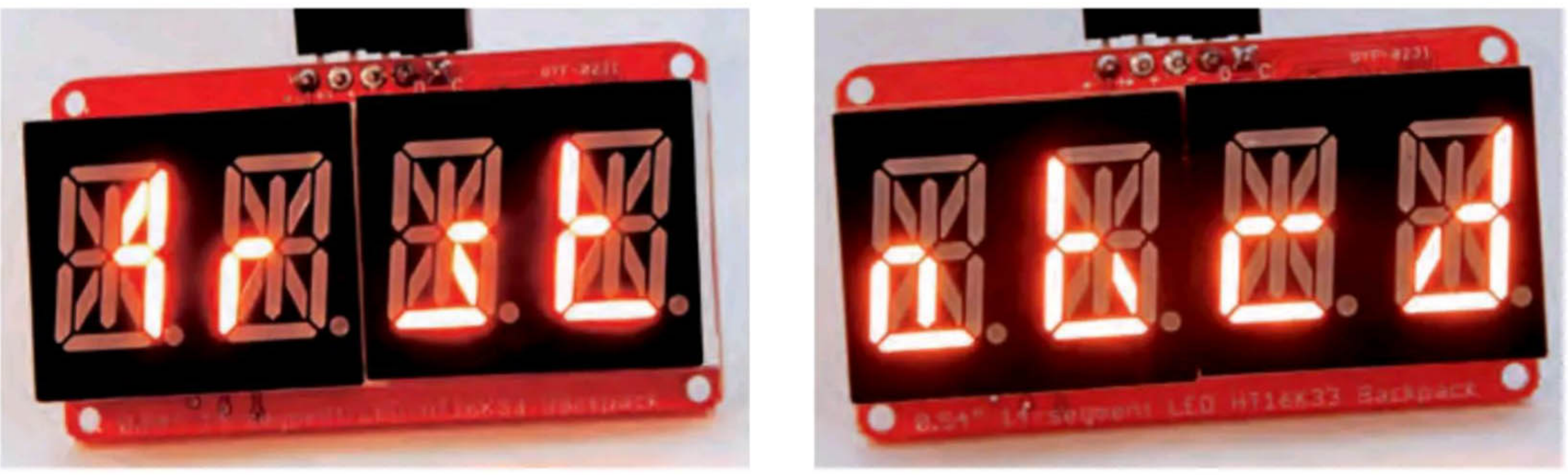

HT16K33
VDD 28
VSS 1
SDA 27
SCL 26
POWER-ON RESET
POR
I2C CONTROLLER
INTERNAL RC OSCILLATOR
TIMING GENERATOR
DISPLAY RAM 16 x 8bits
KEY DATA RAM 13 x 3bits
COMMON SCAN OUTPUT, KEY SCAN OUTPUT, DEVICE ADDRESS SOURCE OUTPUT
ROW DRIVER OUTPUT, INTERRUPT FN OUTPUT, KEY DATA INPUT, DEVICE ADDR DATA INPUT
A(2:0)
9 COM7
8 COM6
7 COM5
6 COM4
5 COM3
4 COM2
3 COM1
2 COM0
25 ROW0 /A2
24 ROW1 /A1
23 ROW2 /A0
22 ROW3
21 ROW4
20 ROW5
19 ROW6
18 ROW7
17 ROW8
16 ROW9
15 ROW10
14 ROW11
13 ROW12
12 ROW13
11 ROW14
10 ROW15

Fig.2: the block diagram for the HT16K33 IC which is used to drive both 14-segment displays.



An example of what the lowercase letters "qrst" and "abcd" look like on the LED module. The letters 'q' and 'a' are some of the more strange choices.

This ability to set the module's I^{2}C address to eight different values means it is possible to connect up to eight of the modules to the same I^{2}C port of an MCU. It also means that if you have another device on your I^{2}C bus within the range of 70h to 77h, you can program the 14-segment display to one of the unused addresses to avoid a collision.

Connecting it to a micro

A nice feature of this module is that its I^{2}C interface makes it easy to connect to most MCUs. This is illustrated in Fig.4, which shows how it can be connected to an Arduino Uno. The module's VHI and VIO pins are both connected to the Arduino's +5V pin, its GND pin to one of the Arduino's GND pins, its SDA pin to the Arduino's A4/SDA pin and its SCL pin to the Arduino's A5/SCL pin.

Note that with R3 and later versions of the Uno, the last two pins can be connected to the SDA and SCL pins at upper left on the Arduino, just to the left of the AREF pin.

Connecting the module to an Arduino Nano is just as easy, as shown in Fig.5. The connections are very similar to those for the Uno in Fig.4.

The only other thing you need to do to get the module to communicate with an MCU is to change its I^{2}C address if necessary; it defaults to 70h when none of the links on the rear of the PCB are joined.

You should find it just as easy to connect the module to most other MCUs, such as a Micromite, Maximite, PicoMite, WebMite and so on.

All that's left then is to come up with some suitable software to drive the display. For an Arduino, as usual, that will involve finding a software library designed to communicate with the HT16K33 module, plus one or more example Arduino sketches to show how it's done.

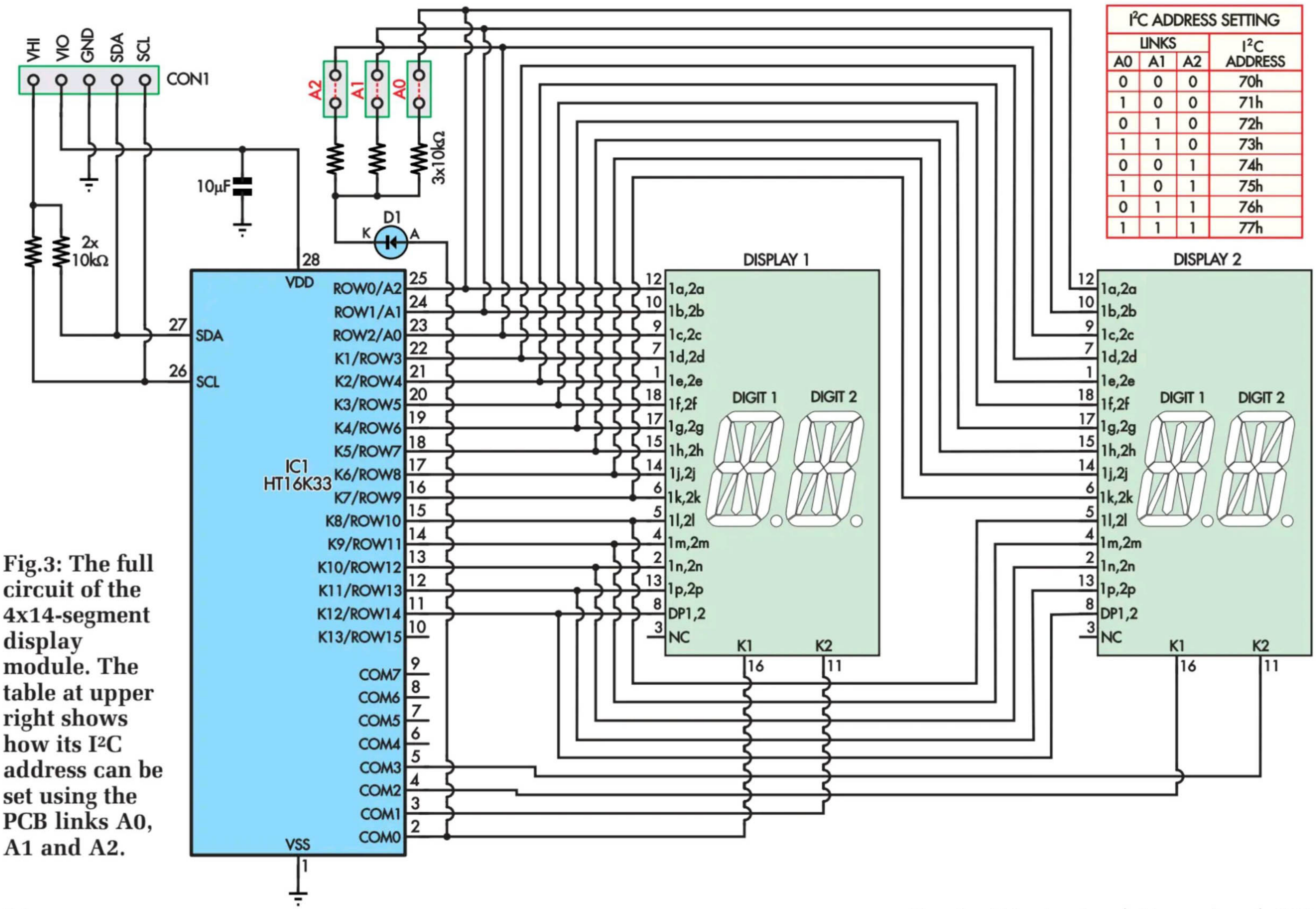
Arduino libraries

After looking around on the web for Arduino libraries written to communicate with the HT16K33 module, the best one I could find was from US firm Adafruit, called Adafruit_LED_Backpack. This one was listed on the main Arduino Reference website but was also available on GitHub:

- **pemag.au/link/abpk**
- **https://github.com/adafruit/Adafruit_LED_Backpack**

However, to work with the 14-segment displays used in this module, two other libraries must be installed: Adafruit-GFX-library and Adafruit_BusIO_library, see:

- **https://github.com/adafruit/Adafruit-GFX-library**
- **https://github.com/adafruit/Adafruit_BusIO**


VHI
VIO
GND
SDA
SCL
CON1
A2
A1
A0
3x10kΩ
10µF
2x 10kΩ
D1
K
A
28
VDD
27
SDA
26
SCL
IC1 HT16K33
VSS
1
ROW0/A2
ROW1/A1
ROW2/A0
K1/ROW3
K2/ROW4
K3/ROW5
K4/ROW6
K5/ROW7
K6/ROW8
K7/ROW9
K8/ROW10
K9/ROW11
K10/ROW12
K11/ROW13
K12/ROW14
K13/ROW15
COM7
COM6
COM5
COM4
COM3
COM2
COM1
COM0
DISPLAY 1
DISPLAY 2
1a,2a
1b,2b
1c,2c
1d,2d
1e,2e
1f,2f
1g,2g
1h,2h
1j,2j
1k,2k
1l,2l
1m,2m
1n,2n
1p,2p
DP1,2
NC
DIGIT 1
DIGIT 2
K1
K2
16
11

I^{2}C ADDRESS SETTING

LINKS			I^{2}C ADDRESS
A0	A1	A2	
0	0	0	70h
1	0	0	71h
0	1	0	72h
1	1	0	73h
0	0	1	74h
1	0	1	75h
0	1	1	76h
1	1	1	77h

Fig.3: The full circuit of the 4x14-segment display module. The table at upper right shows how its I^{2}C address can be set using the PCB links A0, A1 and A2.

Once the three Adafruit libraries have been downloaded (as zip files) and installed on your PC as part of the Arduino IDE or installed via the Library Manager, you will find a "quadalphanum.ino" sketch in the Examples folder. Verify and compile this sketch, then upload it to the Arduino connected to the module, and you should find the module's displays will spring to life.

First, it will show a stream of all the characters it can display (this takes a while). Then, if you have the IDE's Serial Monitor open, it will allow you to type in any combination of four characters you want and they will be displayed immediately. You can repeat this over and over.

While doing this, I took a few photos to illustrate how the module's displays show many of the common alphabetic characters. They should give you a good idea of what can be achieved.

The upper-case characters are all reasonably clear, but the lower-case characters are less so. Some are pretty unclear, like "p" and "q", while some of the symbols are very clear, such as "+" and "-". Unsurprisingly, the numerals are also quite clear.

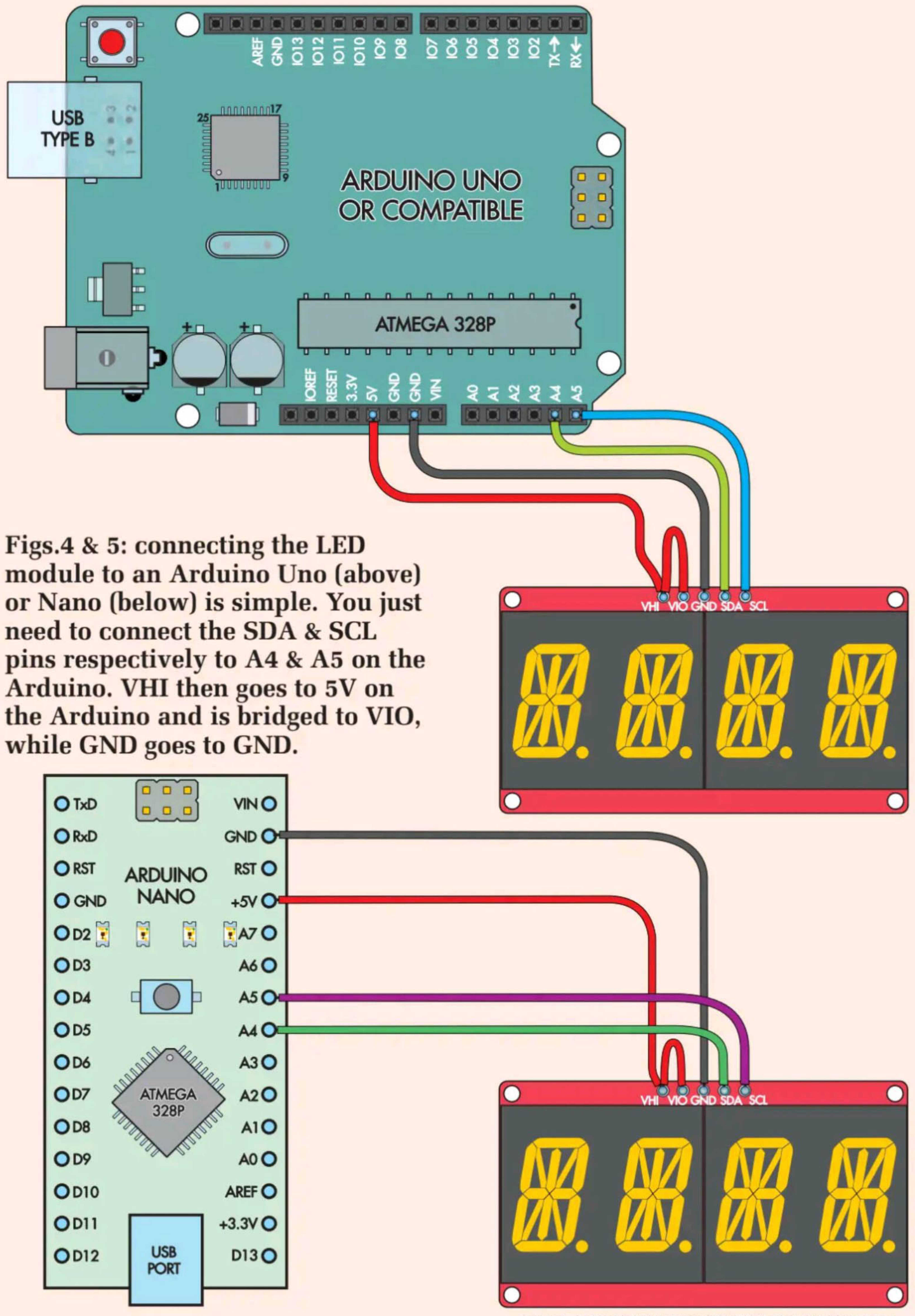
It was a little disappointing to find nothing in the Adafruit libraries to show how to control the light output of the module's displays. However, if you read Holtek's data sheet on the HT16K33 (see pemag.au/link/abpj), they provide quite a bit of information on how the PWM dimming of the displays works and can be achieved.

Editor's note: some lower-case letters could be made more clear by modifying the libraries to change which segments are used. To do this, edit the entries in the "alphafonttable" array that's inside the "Adafruit_LEDBackpack.cpp" file. Examples of shapes we think would be more clear are shown in Fig.6.

Where you can get it

The module shown in the pictures is currently available from a few sellers on AliExpress (eg, pemag.au/link/ac1f & pemag.au/link/ac1g) for around £2 each. It is also available from AdaFruit (ID 2158) for US$10.50.

Adafruit also has versions with different display colours, such as red (1911) for US$9.95, blue (1912), white (2157) or green (2160) displays, each for US$13.95. PE


USB
TYPE B
AREF
GND
IO13
IO12
IO11
IO10
IO9
IO8
IO7
IO6
IO5
IO4
IO3
IO2
TX→
RX←
25
17
9
ARDUINO UNO
OR COMPATIBLE
ATMEGA 328P
IOREF
RESET
3.3V
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5
VHI
VIO
GND
SDA
SCL
TxD
RxD
RST
GND
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
ARDUINO
NANO
ATMEGA
328P
USB
PORT
VIN
GND
RST
+5V
A7
A6
A5
A4
A3
A2
A1
A0
AREF
+3.3V
D13
QUAD ALPHANUMERIC I2C LED DISPLAY MODULE

Figs.4 & 5: connecting the LED module to an Arduino Uno (above) or Nano (below) is simple. You just need to connect the SDA & SCL pins respectively to A4 & A5 on the Arduino. VHI then goes to 5V on the Arduino and is bridged to VIO, while GND goes to GND.

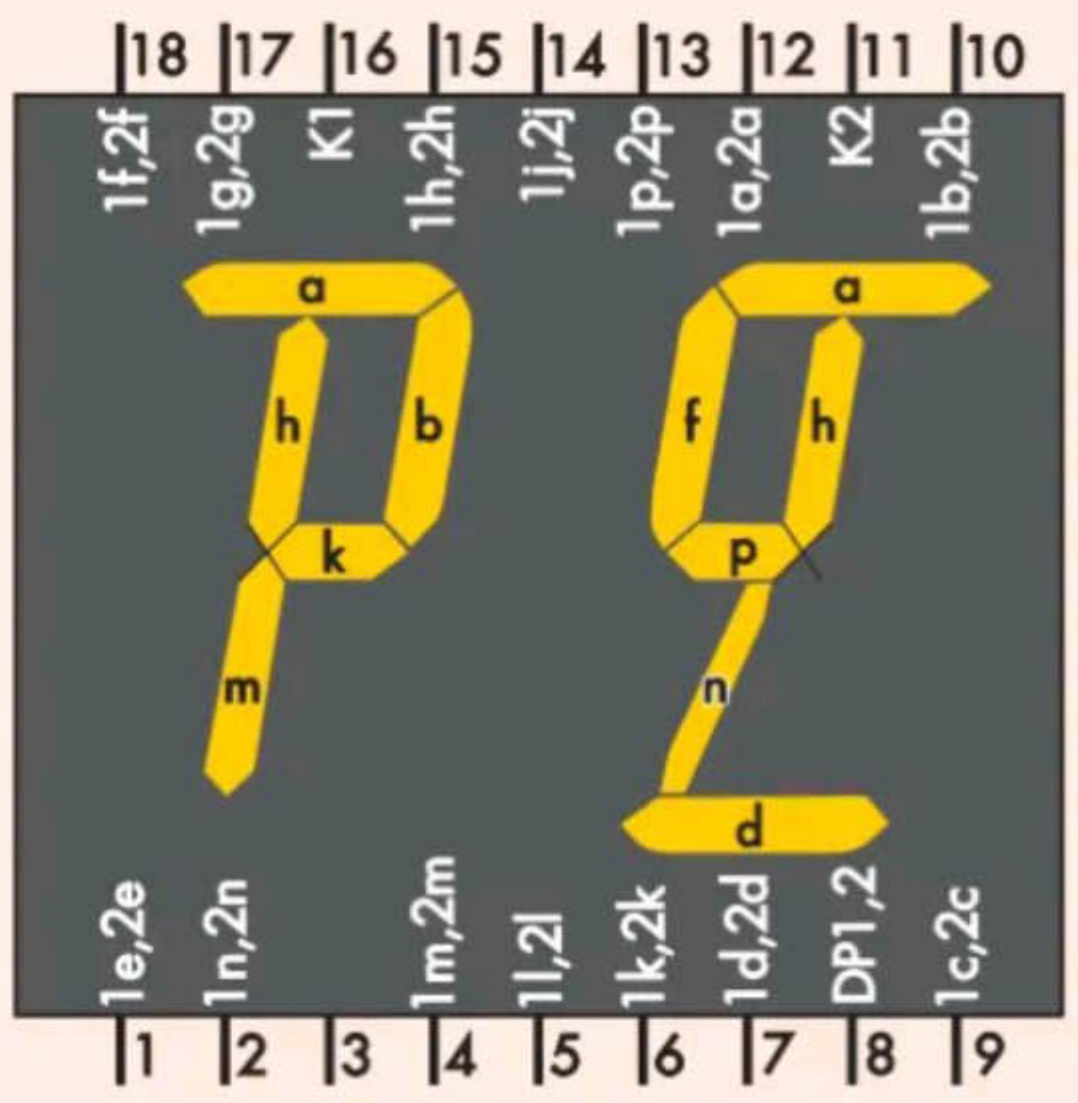

18
17
16
15
14
13
12
11
10
1f,2f
1g,2g
K1
1h,2h
1j,2j
1p,2p
1a,2a
K2
1b,2b
a
h
b
k
m
a
f
h
p
n
d
1e,2e
1n,2n
1m,2m
1l,2l
1k,2k
1d,2d
DP1,2
1c,2c
1
2
3
4
5
6
7
8
9

Fig.6 (left): you can edit the library code to output arguably better representations of different letters. An example of what segments could be enabled for the letters 'p' and 'q' are shown here.

Upper case vs lower case

We recently came across an interesting fact about where the terms "upper case" (capital letters) and "lower case" (smaller letters) came from.

In early printing presses, the "moveable type" letters were kept in cases near the press. As the smaller letters were used more often, they were kept in a box (case) closer to the worker. The capital letters were in a case that was higher and further away, above the other. Hence, "upper case" and "lower case" refer to where the letters were found in those early presses!

Practical Electronics PCB SERVICE

PROJECT	CODE	PRICE
NOVEMBER 2024		
Ideal Diode Bridge Rectifiers (28mm square)	18101241	3.99
Ideal Diode Bridge Rectifiers (22mm square)	18101242	3.99
Ideal Diode Bridge Rectifiers (4-pin SIL)	18101243	3.99
Ideal Diode Bridge Rectifiers (tiny SOT-23)	18101244	3.99
Ideal Diode Bridge Rectifiers (standalone SMD)	18101245	4.99
Ideal Diode Bridge Rectifiers (standalone TH; 2oz copper)	18101246	6.99
Multi-channel Volume Control (volume PCB)	01111221	4.99
Multi-channel Volume Control (control PCB)	01111222	4.99
Multi-channel Volume Control (OLED PCB, black)	01111223	4.99
OCTOBER 2024		
TQFP-32 Programming Adaptor	24108231	4.99
TQFP-44 Programming Adaptor	24108232	4.99
TQFP-48 Programming Adaptor	24108233	4.99
TQFP-64 Programming Adaptor	24108234	4.99
2m VHF FM Test Signal Generator	06107231	4.99
SEPTEMBER 2024		
Calibrated Microphone (SMD version)	01108231	3.99
Calibrated Microphone (TH version)	01108232	3.99
PIC Programming Adaptor	24105231	4.99
Salad Bowl Speaker crossover	01109231	7.99
AUGUST 2024		
Arduino ESR Meter combined L/C-ESR	04106182	9.99
Arduino ESR Meter separate ESR	04106183	7.99
Watering System Controller	15110231	11.99
JULY 2024		
Dynamic NFC Tag (22 × 31mm)	06101231	5.95
Dynamic NFC Tag (86 × 54mm)	06101232	7.95
Reciprocal Frequency Counter	CSE230101C	8.95
Reciprocal Frequency Counter front panel (black mask)	CSE230102	7.95
Pi Pico-based Thermal Camera	04105231	6.95
JUNE 2024		
Loudspeaker Test Jig	04106231	12.95
MAY 2024		
GPS-Disciplined Oscillator	04103231	12.95
Dual RF Amplifier	CSE220602A	5.95
Songbird	08103231	7.95
APRIL 2024		
Basic RF Signal Generator	CSE221001	8.99
Basic RF Signal Generator front panel (black solder mask)	CSE220902B	9.99
Silicon Chirp Cricket	08101231	7.99
Test Bench 'Swiss Army Knife'	04110221	9.99
MARCH 2024		
Digital Volume Control Potentiometer SMD version	01101231	8.95
Digital Volume Control Potentiometer through-hole version	01101232	9.95
Advanced SMD Test Tweezers body	04106221	6.95
Advanced SMD Test Tweezers legs (pair)	04106212	7.95
FEBRUARY 2024		
Active Mains Soft Starter	10110221	10.95
JANUARY 2024		
Q Meter	CSE220701	8.95
Q Meter front panel (black solder mask)	CSE220704	7.95
Raspberry Pi Pico W BackPack	07101221	8.95

PROJECT	CODE	PRICE
DECEMBER 2023		
Digital Boost Regulator	24110224	8.95
Dual-Channel Power Supply for Breadboards	04112221	11.95
Display Adaptor for the Breadboard PSU	04112222	9.95
NOVEMBER 2023		
LC Meter Mk3	CSE220503C	9.95
DC Supply Filter for vehicles	08108221	8.95
Discrete Audio Op Amp PCB	AO1-JUL23	9.95
Discrete Audio Op Amp PCB + essential components	AO2-JUL23	17.95
OCTOBER 2023		
Buck/Boost Charger Adaptor	14108221	9.95
PIC Breakout Board for SOIC parts	24110225	5.95
PIC Breakout Board for DIP parts	24110222	5.95
AVR64DD32 Breakout board	24110223	5.95
Automatic Train Controller	09109221	6.95
Chuff Sound module	09109222	6.95
SEPTEMBER 2023		
Mini LED Driver	16106221	7.95
New GPS-Synchronised Clock	19109221	9.95
AUGUST 2023		
110dB RF Attenuator	CSE211003	9.95
Universal Battery Charge Controller (2023 update)	14107192	9.95
Wide-Range OhmMeter	04109221	12.95
JULY 2023		
Multimeter Checker/Calibrator	04107221	10.95
MIDI Spectral Sound Synthesiser (full kit – see p.25)	N/A	N/A
JUNE 2023		
Arduino Programmable Load	04105221	8.95
Buck-Boost LED Driver	16103221	8.95
MAY 2023		
Precision AM-FM DDS Signal Generator	CSE211002	14.95
Improved SMD Test Tweezers programmed PIC	0410621PIC	12.95
AO 2x-dual-to-quad through-hole (pack of five)	AO1-MAY23	6.95
AO 2x-dual-to-quad SMD (pack of five)	AO2-MAY23	6.95
AO 2x-single-to-dual through-hole (pack of five)	AO3-MAY23	6.95
AO 2x-single-to-dual SMD (pack of five)	AO4-MAY23	6.95
APRIL 2023		
500W Amplifier Module	see p.22, April 2023	
Clipping Indicator (per channel)	01112211	7.95
CD Welder Power Supply (one needed)	29103221	9.95
CD Welder Controller (one needed)	29103222	9.95
CD Welder Energy Storage module (several needed)	29103223	7.95
AO Universal Dual Op Amp Board	AO1-APR23	9.95
AO Stereo RIAA precision passives kit for dual op amp	AO2-APR23	8.95
MARCH 2023		
Pico BackPack	07101221	9.95
Semaphore Signal (controller)	09103221	7.95
Semaphore Signal (blade)	09103222	5.95
FEBRUARY 2023		
Tesla Coil driver board	26102221	9.95
Tesla Coil potentiometer board	26102222	5.95
Cooling Fan Controller & Loudspeaker Protector	01102221	8.95
Remote Gate Controller	11009121	12.95

PCBs for most recent *PE/EPE* constructional projects are available. From the July 2013 issue onwards, PCBs with eight-digit codes have silk screen overlays and, where applicable, are double-sided, have plated-through holes, and solder mask. They are similar to photos in the project articles. Earlier PCBs are likely to be more basic and may not include silk screen overlay, be single-sided, lack plated-through holes and solder mask.

Always check price and availability in the latest issue or online. A large number of older boards are listed for ordering on our website.

In most cases we do not supply kits or components for our projects. For older projects it is important to check the availability of all components before purchasing PCBs.

Back issues of articles are available – see Back Issues page for details.

Double-sided | plated-through holes | solder mask

PROJECT	CODE	PRICE
JANUARY 2023		
Classic LED Metronome – 8-LED	23111211	7.95
Classic LED Metronome – 10-LED	23111212	8.95
Multi-Channel Speaker Protector – 6 channel	01101221	9.95
Multi-Channel Speaker Protector – 4 channel	01101222	7.95
Remote Control Range Extender – IR-to-UHF	15109212	5.95
Remote Control Range Extender – UHF-to-IR	15109211	7.95
AO Universal Single Op Amp Board	AO1-JAN23	7.95
DECEMBER 2022		
Hummingbird Amplifier	01111211	9.95
SMD Trainer PCB	29106211A	8.95
SMD Trainer PCB + parts	29106211B	13.95
NOVEMBER 2022		
USB Cable tester – main PCB	04108211	12.95
USB Cable tester – front panel	04108212	5.95
USB Cable tester – optional panel	SC5970	5.95
Model Railway Carriage Lights – PCB	09109211	6.95
AO transfmr PCB – standard VTX-A range	VTX-101-007	6.95
AO transfmr PCB – dual-outline VTX102-3001/101-3002	VTX-Dual	6.95
OCTOBER 2022		
SMD Test Tweezers – PCB and pair of tweezer arms	04106211-2	11.95
SMD Test Tweezers – programmed PIC12F1572-I/SN	0410621A	7.95
Tele-com	12110211	12.95
SEPTEMBER 2022		
Touchscreen Digital Preamp – main board	01103191	12.95
Touchscreen Digital Preamp – adaptor board pair	01103192	5.95
20A DC Motor Speed Controller	11006211	9.95
AUGUST 2022		
Multi-purpose Battery Manager – I/O Expander module	11104212	5.95
Multi-purpose Battery Manager – Switch Module	11104211	8.95
Simple MIDI Music Keyboard (for 8 switches)	23101213	6.95
Nano Pong	08105212	7.95
JULY 2022		
Silicon Labs AM/FM/SW Radio	CSE210301C	10.95
Level Crossing Controller	09108211	6.95
JUNE 2022		
Full-wave Motor Speed Controller	1010221	8.95
PIC Programming Helper for 8-pin PICs only	24106211	7.95
PIC Programming Helper for 8, 14 or 20-pin PICs	24106212	10.95
Advanced GPS Computer	05102211	9.95
MAY 2022		
Bus board PCB for Analogue Vocoder	AO1-MAY22	10.95
Complete set of 14 PCBs for Analogue Vocoder	AO2-MAY22	97.95
Programmed EEPROM for Digital FX Unit	FX1-MAY22	10.95
Programmed PIC for Digital FX Unit using potentiometer	FX2-MAY22	8.95
APRIL 2022		
64-key MIDI Matrix shield	23101211	8.95
64-key MIDI Matrix switch board	23101212	11.95
High-current Battery Balancer	14102211	10.95
Digital FX Unit – using potentiometer	01102211	9.95
Digital FX Unit – using BCD switch	01102212	9.95
Universal Audio PSU	AO1-APR22	11.95
MARCH 2022		
Mini Isolated Serial Link	24102211	£5.95
Busy Loo Indicator	16112201	£5.95
Analogue Vocoder – Band-pass filter board	AO1-MAR22	9.95
Analogue Vocoder – HP/LP filter board	AO2-MAR22	9.95

PROJECT	CODE	PRICE
FEBRUARY 2022		
Arduino-based Power Supply	18106201	9.95
Battery Monitor Logger	11106201	10.95
Electronic Wind Chimes	23011201	10.95
Analogue Vocoder – Driver Amplifier	AO-FEB22	8.95
JANUARY 2022		
Vintage battery Radio Li-ion Power Supply	11111201	9.95
MiniHeart: A Miniature Heartbeat Simulator	01109201	8.95
DECEMBER 2021		
AM/FM/SW Digital Receiver	CSE200902A	13.95
Balanced Input and Attenuator for USB CODEC	01106202	11.95
NOVEMBER 2021		
Dual Battery Lifesaver	11111202	£6.95
OCTOBER 2021		
Mini Wi-Fi LCD BackPack	24106201	£8.95

For the many pre-2020 PCBs that we stock please see the PE website: www.electronpublishing.com

PE/EPE PCB SERVICE

Order Code	Project	Quantity	Price

Name ....

Address ....

....

Tel ....

Email ....

Card No ....

Valid From ....Expiry Date ....

Card Security No ....

You can also order PCBs by phone, email or via the shop on our website: www.electronpublishing.com

No need to cut your issue – a copy of this form is just as good!

All prices include VAT and UK p&p. Add £4 per *project* for post to Europe; £5 per *project* outside Europe.

Orders and payment should be sent to:

Practical Electronics, Electron Publishing Ltd
113 Lynwood Drive, Merley, Wimborne, Dorset BH21 1UU
Tel 01202 880299 Email: pe@pemag.au
On-line Shop: www.electronpublishing.com

Sorry, cheques cannot be accepted as we are based in Australia and our UK bank account does not allow cheque deposits.

NOTE: Most boards are in stock and sent within seven days of order receipt. Please allow up to 28 days delivery if we need to restock.

Practical Electronics PCB SERVICE

PROJECT	CODE	PRICE
SEPTEMBER 2021		
USB SuperCodec PCB	01106201	£14.95
Audio DDS Oscillator PCB	01110201	£5.95
Audio DDS Oscillator rotary encoder	01110201-ENC	6.95
Programming Adaptor Board for Audio DDS Oscillator	01110202	£5.95
High-power Ultrasonic Cleaner main PCB	04105201	£14.95
High-power Ultrasonic Cleaner front-panel PCB	04105202	
Night Keeper Lighthouse PCB	08110201	£5.95
AUGUST 2021		
Ol' Timer PCB	19104201	£11.95
Ol' Timer 8x8 RGB LED module using WS2812B	19104201-88	£8.95
Ol' Timer set of acrylic case pieces and spacer	19104201-ACR	£8.75
Ol' Timer DS3231 RTC IC wide SOIC-16	19104201-RTC	£5.95
Wideband Digital RF Power Meter	04106201	£9.75
Switchmode 78xx regulators ***(PACK of 5!)***	18105201	£7.95
Cool Beans SMAD display	CB-AUG21	£11.95
JULY 2021		
ATtiny816 Breakout / Dev Board with Capacitive Touch	24110181	£9.75
IR Remote Control Assistant (Jaycar version)	15005201	£8.95
IR Remote Control Assistant (Altronics version)	15005202	£8.95
PIC18F Development Board	PNM-JUL21	£12.95
Microphone Preamplifier	AO-JUL21	£11.95
JUNE 2021		
Roadies' Test Signal Generator (surface-mount version)	01005201	£8.95
Roadies' Test Signal Generator (through-hole version)	01005202	£9.95
Touchscreen Wide-range RCL Box (Resistor module)	04104201	£18.95
Touchscreen Wide-range RCL Box (Ind/Cap module)	04104202	
KickStart Part 3 – Gyrator-based Audio Filter	KS3-2021	£7.95
MAY 2021		
7-Band Equaliser (Mono)	01104201	£8.95
7-Band Equaliser (Stereo)	01104202	£10.95
Car Altimeter	05105201	£7.95
APRIL 2021		
Reflow Oven – DSP Active Crossover (CPU)	01106193	£19.95
Reflow Oven – DSP Active Crossover (Front panel)	01106195	
Reflow Oven – DSP Active Crossover (LCD)	01106196	
Frequency Reference Signal Distributor	CSE200103	£8.95
MARCH 2021		
Nutube Guitar Effects Pedal	01102201	£12.95
Programmable Thermal Regulator (Peltier Interface)	21109181	£18.95
Programmable Thermal Regulator (Peltier Driver)	21109182	
Tunable HF Preamp	CSE190502	£8.95
FEBRUARY 2021		
4G Remote Monitoring	27111191	£9.95
JANUARY 2021		
Nutube Valve Preamplifier	01112191	£12.95
Arduino DCC Controller	09207181	£10.95
DECEMBER 2020		
Pseudo-Random Sequence Generator	16106191	£7.95
Clever Charger	14107191	£11.95
PE Theremin Amplifier	AO-1220-01	£8.95

PROJECT	CODE	PRICE
NOVEMBER 2020		
LED Christmas Tree (1 off)	16107181-1	£6.95
LED Christmas Tree (4 off)	16107181-2	£14.95
LED Christmas Tree (12 off)	16107181-3	£24.95
LED Christmas Tree (20 off)	16107181-4	£34.95
USB/SPI Interface Board	16107182	£5.95
45V/8A Power Supply PCB *plus* acrylic spacer	18111181	£14.95
45V/8A Power Supply front panel five-way display bezel	18111181-BZ	£3.95
Five-way LCD Panel Meter/Display	18111182	£7.95
OCTOBER 2020		
Digital Audio Millivoltmeter	04108191	£9.95
Precision Signal Amplifier	04107191	£7.95
SEPTEMBER 2020		
PE Theremin PSU	AO-0920-01	£5.95
PE Theremin PSU transformer	AO-0920-02	£7.95
Micromite Explore-28	07108191	£6.95
Ultrabrite LED Driver	16109191	£6.95
AUGUST 2020		
Micromite LCD BackPack V3	07106191	£9.95
Steering Wheel Audio Button to Infrared Adaptor	05105191	£7.95
JULY 2020		
AM/FM/CW Scanning HF/VHF RF Signal Generator	04106191	£13.95
Speech Synthesiser with the Raspberry Pi Zero	01106191	£5.95
PE Mini-organ PCB	AO-0720-01	£14.95
PE Mini-organ selected parts	AO-0720-02	£8.95
High-current Solid-state 12V Battery Isolator – control	05106191	£6.95
High-current Solid-state 12V Battery Isolator FET (2oz)	05106192	£9.95
JUNE 2020		
Arduino breakout board – 3.5-inch LCD Display	24111181	£6.95
Six-input Audio Selector main board	01110191	£10.95
Six-input Audio Selector switch panel board	01110192	
MAY 2020		
Ultra-low-distortion Preamplifier Input Selector	01111112	£11.25
Ultra-low-distortion Preamplifier pushbutton Input Selector	01111113	
Universal Regulator	18103111	£7.95
433MHz Wireless Data Repeater	15004191	£8.50
Bridge-mode Adaptor for Amplifier	01105191	£7.95
iCEstick VGA Terminal	02103191	£5.95
Analogue noise with tilt control	AO-0520-01	£7.95
Audio Spectrum Analyser	PM-0520-01	£8.95
APRIL 2020		
Flip-dot Display black coil board	19111181	£14.95
Flip-dot Display black pixels	19111182	
Flip-dot Display black frame	19111183	
Flip-dot Display green driver board	19111184	
MARCH 2020		
Diode Curve Plotter	04112181	£10.95
Steam Train Whistle / Diesel Horn Sound Generator	09106181	£8.50
Universal Passive Crossover (one off)	UPC0320	£12.50
FEBRUARY 2020		
Motion-Sensing 12V Power Switch	05102191	£5.95
USB Keyboard / Mouse Adaptor	24311181	£8.50
DSP Active Crossover (ADC)	01106191	£29.95
DSP Active Crossover (DAC) ×2	01106192	
DSP Active Crossover (CPU)	01106193	
DSP Active Crossover (Power/routing)	01106194	
DSP Active Crossover (Front panel)	01106195	
DSP Active Crossover (LCD)	01106196	

CLASSIFIED ADVERTISING

Practical Electronics

If you want your advertisements to be seen by the largest readership at the most economical price then our classified page offers excellent value.

The rate for semi-display space is £10 per centimetre high, with a minimum height of 2·5cm. All semi-display advertisements have a width of 5.5cm. The prepaid rate for classified adverts is 40p per word (minimum 12 words).

For rates, further information on display and classified advertising or to book an ad, please email us at pe@pemag.au

Practical Electronics reaches more UK readers than any other UK monthly hobby electronics magazine. Our sales figures prove it.

We have been the leading monthly magazine in this market for the last twenty-seven years.

BOWOOD ELECTRONICS LTD
Suppliers of Electronic Components
www.bowood-electronics.co.uk
Unit 10, Boythorpe Business Park, Dock Walk, Chesterfield,
Derbyshire S40 2QR. Sales: 01246 200 222
Send large letter stamp for Catalogue

Electrical Industries Charity (EIC)
We help people working in the electrical, electronics and energy community as well as their family members and retirees.
We use workplace programmes that give the industry access to financial grants and a comprehensive range of free and confidential services.
www.electricalcharity.org
electrical industries charity

COAST ELECTRONICS
BREAKOUTS-COMPONENTS-
CONTRACT DESIGN-3D PRINTER PARTS-
MUSICAL-MICROCONTROLLERS
WWW.COASTELECTRONICS.CO.UK

MISCELLANEOUS

VALVES AND ALLIED COMPONENTS?
For free stock list and/or advice, please contact me: geoffdavies337@gmail.com Telephone: 01788 574774

PIC DEVELOPMENT KITS, DTMF kits and modules, CTCSS Encoder and Decoder/Display kits.
Visit www.cstech.co.uk

COMPONENTS AS JOB LOT
Large stock. Transistors, Diodes, Logic, Memory and Analogue parts. 1000s of passive items. Catalogued and sorted. 'New Old Stock' in racks.
Email: rojobarn@gmail.com

Andrew Kenny – Qualified Patent Agent
EPO UKIPO USPTO
Circuits Electric Machinery Mechatronics
Web: www.akennypatentm.com
Email: Enquiries@akennypatentm.com
Tel: 0789 606 9725

ADVERTISING INDEX

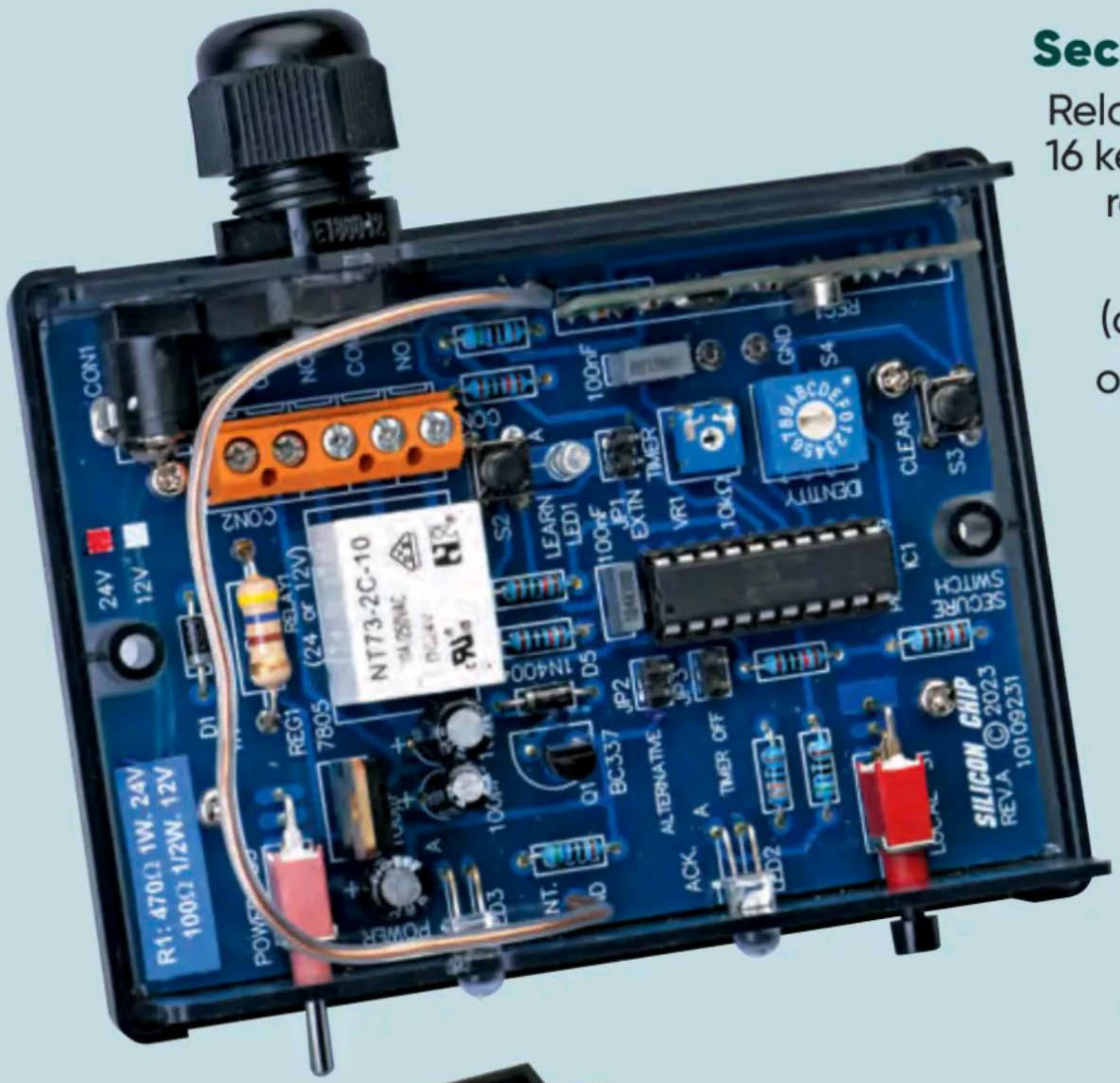
FLOWCODE . . . . Outer back cover
HAMMOND ELECTRONICS Ltd. 64
MICROCHIP TECHNOLOGY . . . . Inner front cover
STEWART OF READING 65

Advertising mailing address
Electron Publishing Australia
PO Box 194
Matraville NSW 2036
Australia

Email pe@pemag.au
Web www.electronpublishing.com

For editorial contact details, see page 2.

Next Month – in the December issue

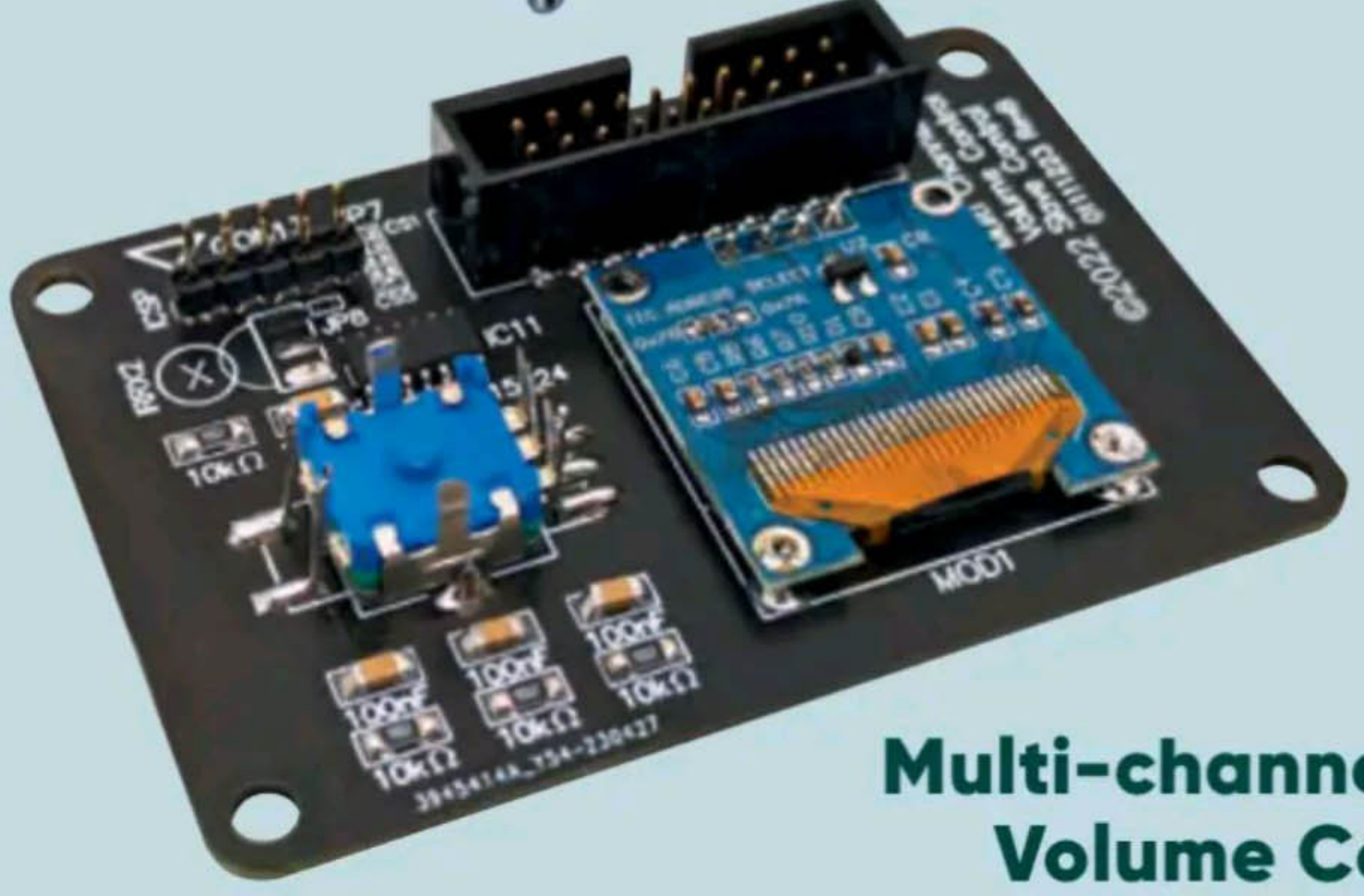


TIMER
ON
OFF
SECURE REMOTE SWITCH
TRANSMITTER →

Secure Remote Switch

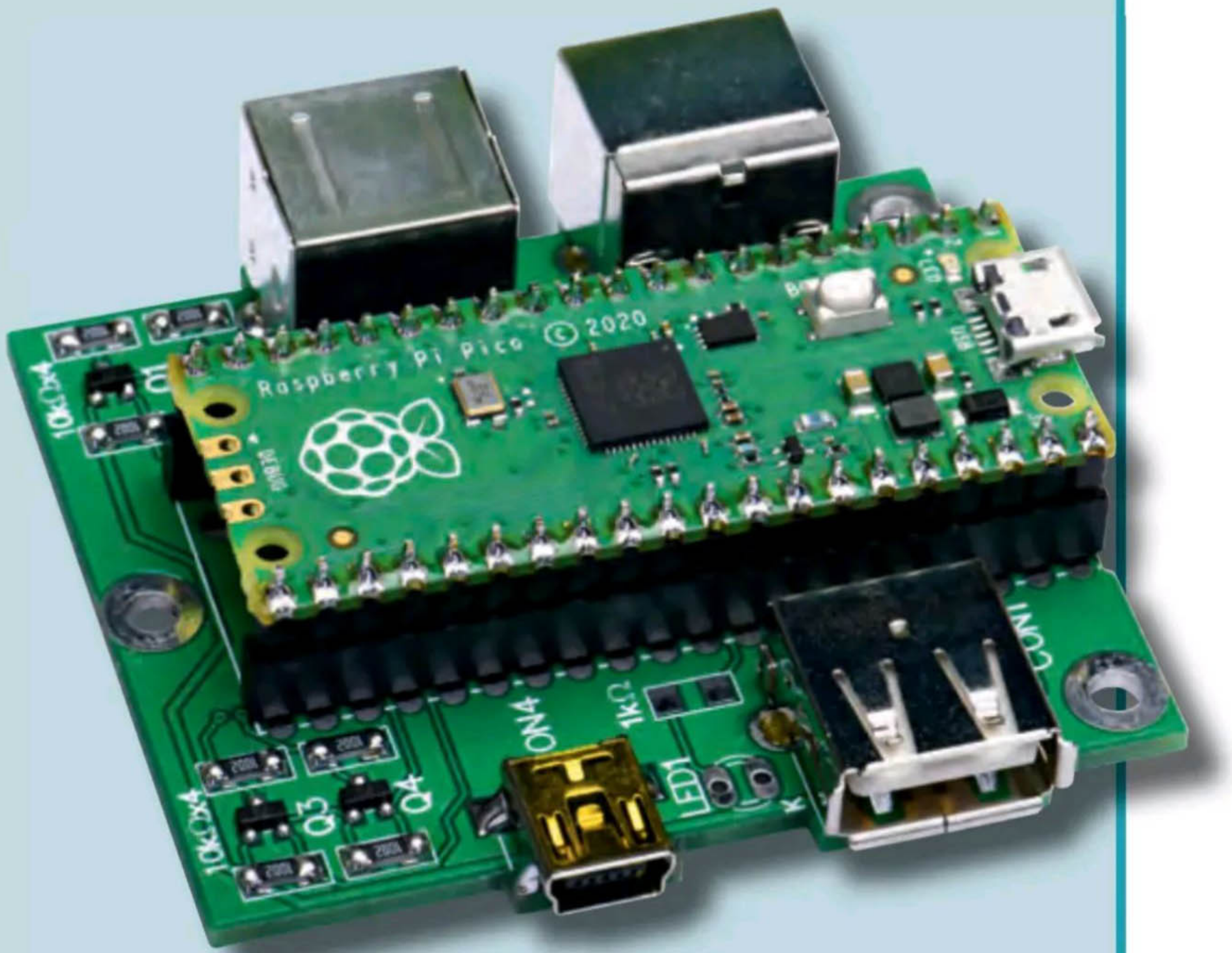
Relay switching with up to 16 keyfob transmitters and rolling code security

(delayed due to lack of space in this issue)



Multi-channel Digital Volume Control

Building it and wiring it up, including the optional rotary encoder module with OLED screen shown here.



Convert USB to PS/2

Use a modern mouse and/or keyboard with legacy devices that only have PS/2 ports.

PLUS! All your favourite regular columns from *Max's Cool Beans* and *Circuit Surgery* to *Audio Out*, *Techno Talk* and *Net Work*.

On sale 4th of November, 2024

Content may be subject to change

Practical Electronics

NEW subscriptions hotline!

We have recently changed the way we sell and renew subscriptions. To subscribe or renew online, paying by credit/debit card or PayPal, please visit our shop at: www.electronpublishing.com

For those wishing to pay by direct debit, cheque or over the telephone, please contact 'Select Publisher Services' by calling 01202 087631, or send a cheque (payable to: 'Practical Electronics') with your details to:

Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Remember, we print the date of the last issue of your current subscription in a box on the address sheet that comes with your copy.

Digital subscribers, please call 01202 880299 or visit: www.electronpublishing.com

Published on approximately the first Thursday of each month by Silicon Chip Publications, U1 234 Harbord Road Brookvale NSW 2100 Australia. Printed in England by The Manson Group, Reynolds House, 8 Porters Wood, Valley Road Industrial Estate, St Albans AL3 6PZ. Distributed by Seymour, 86 Newman St, London W1T 3EX. Subscriptions UK: £33.99 (6mo); £59.99 (12mo.); £114.99 (2yrs). EUROPE: airmail service, £38.99 (6mo); £69.99 (12mo); £129.99 (2yrs). REST OF THE WORLD: airmail service, £46.99 (6mo); £84.99 (12mo); £164.99 (2yrs). Make payments to 'Practical Electronics', Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH United Kingdom. Email: pesubs@selectps.com. PRACTICAL ELECTRONICS is sold subject to the conditions that it shall not, without the written consent of the Publishers first having been given, be lent, resold, hired out or otherwise disposed of by way of Trade at more than the recommended selling price on the cover, and that it shall not be lent, resold, hired out or otherwise disposed of in a mutilated condition or in any unauthorised cover by way of Trade or affixed to or as part of any publication or advertising, literary or pictorial matter.

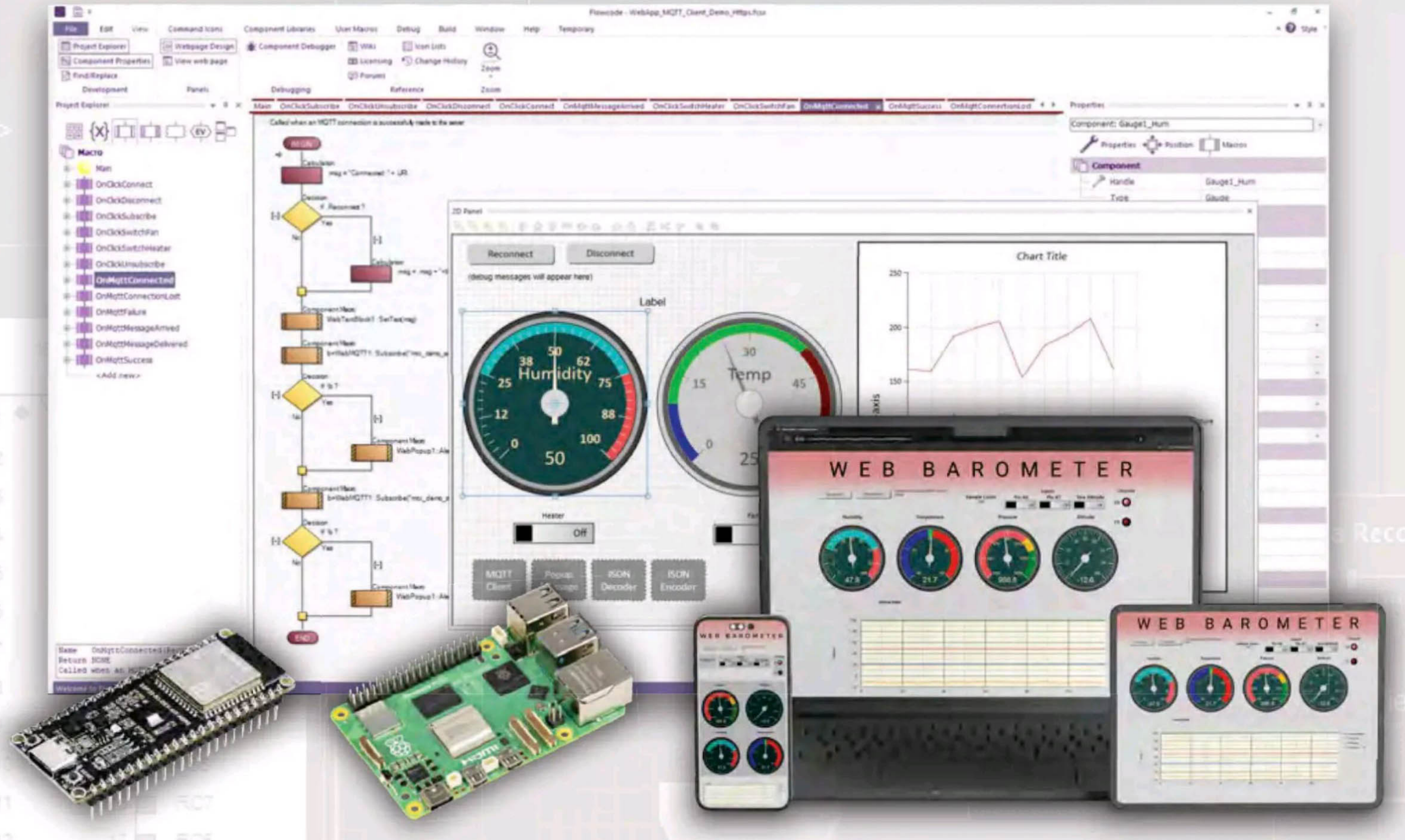
Did you know our online shop now sells the current issue of *PE* for £5.99 inc. p&p?

Practical Electronics
The UK's premier electronics, computing and maker magazine
Circuit Surgery
DAC frequency response correction
Teach-In 2024
Learn electronics with the ESP32, Part 9 – Putting it all together
Practically Speaking
Avoiding electric shocks during repairs
Six PCBs for highly efficient active rectification
Control the volume of up to 20 audio channels with a touchscreen or remote!
PLUS!
MG4 XPower EV review by Julian Edgar
Techno Talk – On robots and ukuleles
Cool Beans – There's more than one way to blank zeros
Net Work – PE's 60th anniversary and its history
Nov 2024 £5.99
www.electronpublishing.com
@practicalelec
practicalelectronics
Audio Out
More on electronic audio switching
2m band VHF Test Signal Generator
Oct 2024 £5.99

You read that right! We now sell the current issue of your favourite electronics magazine for exactly the same price as in the High Street, but we deliver it straight to your door – and for UK addresses we pay the postage. No need to journey into town to queue outside the newsagent. Just go to our website, set up an account in 30 seconds, order your magazine and we'll do the rest.

www.electronpublishing.com

FLOWCODE


Reconnect
Disconnect
Label
Humidity
Temp
Chart Title
WEB BAROMETER
WEB BAROMETER

CREATE HIGHLY FUNCTIONAL ELECTRONIC SYSTEMS USING LOW COST HARDWARE, LIKE ESP32 AND RPI, TO CREATE FANTASTIC HUMAN MACHINE INTERFACES FOR YOUR PROJECTS AND MOBILE PHONES.

Flowcode hobby for ESP32 is FREE
Flowcode Pro with ESP32 pack £340 +VAT

10% off for PE readers using code: **WEB10**

FLOWCODE 10.1 WITH MORE THAN 20 MICROCONTROLLER TARGETS INCLUDING ARDUINO UNO, PIC18877, RPI2B, AND ESP32 IS NOW FREE FOR HOBBYISTS.

☎ +44 (0) 1422 252 380 — ✉ sales@matrixtsl.com

www.flowcode.co.uk